UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Escola d'Enginyeria de Telecomunicació
i Aeroespacial de Castelldefels

AirplaneSolutions

TREBALL DE FI DE GRAU

TFG TITLE: Research in ATM Procedures and En-Route Charges

DEGREE: Aerospace Systems Degree

AUTHOR: Anna Ruiz Ayala

ADVISOR: Cristina Barrado Muxi

SUPERVISOR: Omar López Ruiz

DATE: July 15, 2019

Títol: Investigació sobre els Procediments del Tràfic Aeri i les Taxes d'Aeronavegació
Autor: Anna Ruiz Ayala

Director: Cristina Barrado Muxi

Supervisor: Omar López Ruiz

Data: 15 de juliol de 2019

Resum

Els nostres cels es troben més congestionats que mai. El tràfic aeri comercial incrementarà el doble els pròxims 15 anys, i per tant, l'espai aeri encara estarà més congestionat. Aviat, hi haurà més de 25,000 vols tripulats volant a qualsevol temps donat. A més, les companyies aèries s'enfronten a una problemàtica: la gestió dels costos operacionals, que provoquen sobrecostos.
Aquest projecte es realitza en col·laboració amb l'empresa Airplane Solutions, la qual per tal de controlar els costos directes i reduir aquest sobrecost, ha introduït una solució que proporciona un càlcul de costos en temps real. L'objectiu d'aquest projecte és desenvolupar una eina la qual analitzi la variació de cost total de la fase en-ruta entre el vol real i el pla de vol, seguint els principis del actual sistema europeu que gestiona els costos de la fase en-ruta, operat per Eurocontrol. Els tres paràmetres principals per calcular el cost de navegació és la distància ortodròmica del vol (el trajecte considerat és el pla de vol), el pes màxim a l'enlairament i el preu unitari per Estat Membre d'Eurocontrol.
Per crear l'espai de treball, primer s'han dissenyat els espais aeris i s'ha desenvolupat un codi de programació amb Python per tal de processar els sectors i els vols, exportats de la aplicació NEST. D'aquesta manera, es poden calcular els costos de la fase en-ruta. Finalment, els resultats, es mostren gràficament en una plataforma web: CARTO. Amb aquesta plataforma es poden veure els resultats en un mapa de manera clara i molt visual.
L' últim punt d'aquest treball és l'anàlisi dels resultats. Primerament, s'ha realitzat un anàlisi global. Seguidament, aquest anàlisi s'ha centrat per companyia aèria i pels països implicats en aquesta taxa. Amb aquest estudi, s'ha demostrat que hi ha una variació de costos entre el pla de vol i la ruta realment realitzada. Globalment, les rutes reals resulten més econòmiques que els plans de vol.
En conclusió, es demostra que hi ha un sobrecost en la taxa de la fase en-ruta i per tant, aquesta eina ajuda a les aerolínies i a la gestió del trànsit aeri a crear un espai aeri més eficient i econòmic.

Title : Research in ATM Procedures and En-Route Charges
Author: Anna Ruiz Ayala

Advisor: Cristina Barrado Muxi

Supervisor: Omar López Ruiz

Date: July 15, 2019

Overview

Our skies are busier than ever. And with commercial air traffic set to double in the next 15 years, they will only get busier. Soon, there will be 25,000 manned flights in the air at any given time. Furthermore, airlines face cost management challenges which derive in overpayment for the airlines. This project is in collaboration with Airplane Solutions company, which in order to control DOCs and reduce this overpayment, has introduced a solution that provides a real-time costs calculation.

The objective of this project is to develop a tool which analyse the total cost variation of the en-route phase between the real flight and the flight plan according to the principles of current Europe En-Route Charge System operated by Eurocontrol. The three principal parameters to calculate the route charge are the great circle distance of the flight (the path used to calculate it is extracted from the flight plan), the MTOM and the Unit Rate per Eurocontrol State Member.

To generate the working environment, first airspaces and a Python script have been developed to process the sectors and the flights, extracted from NEST application, in order to compute the en-route charges. Afterwards, the results are plotted in CARTO platform, which allows to see the results on a map in a very clearly visual display.

At last, an analysis of the results is performed. In this case, a global analysis has been done first. Next, the analysis is focused by airline and involved European countries. With this exhaustive analysis it can be demonstrated that there is a cost variation between the flight plan and the real route, emphasizing that in global, the real routes suppose benefits in en-route charges because these charges based on flight plans are high-priced.

In conclusion, there is an extra cost in the en-route charges and so on, this project helps airlines and the air traffic management to build a more efficient and economic airspace.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ACRONYMS

ATM Air Traffic Management

ICAO International Civil Aviation Organization

IFR Instrument Flight Rules

DOCs Direct Operational Costs

ATC Air Traffic Controllers

APS Airplane Solutions Company

SWIM System Wide Information Management

SESAR Single European Sky ATM Research

EUROCONTROL European Organisation for the Safety of Air Navigation

CRCO Central Route Charges Office

ECAC European Civil Aviation Conference

SES Single European Sky

EU European Union

NM Network Manager

MTOW/MTOM Maximum Take-Off Weight/Mass

FIR Flight Information Region

UIR Upper Information Region

FL Flight Level

AIP Aeronautical Information Publication

SaaS Software as a service

NEST Network strategic modelling tool

BAW British Airways

DLH Lufthansa

EWG Eurowings

EZY EasyJet

VLG Vueling

INTRODUCTION

As the world economy grows, so does air traffic and airspace congestion. This expansion puts increasing pressure on aviation infrastructure, systems and facilities.

According to the Global Market Forecast from Airbus [1], a long-term growth potential for the aviation industry is confirmed:

- The commercial aviation Industry has been resilient to external shocks, traffic has doubled since 2000.
- In the next 15 years, is expected that traffic will double.
- An average of 4.4% increment in traffic growth passenger is also expected.

During 2018, global passenger traffic exceeded the 4.3 billion passengers who travelled on scheduled flights, 6.1% more than in 2017; according to the preliminary report of the International Civil Aviation Organization (ICAO)[1]. The number of departures increased to 38 million worldwide and world traffic in terms of number of passengers/kilometers transported (PKT) grew by 6.7% to 8.2 billion.

Eurocontrol published a report [2] presenting the update of the EUROCONTROL 20-year forecast for Instrumental Flight Rules (IFR)[2] flight movements in Europe up to 2040. This report uses simulations to prove an evident air traffic growth in Europe over the next few years:

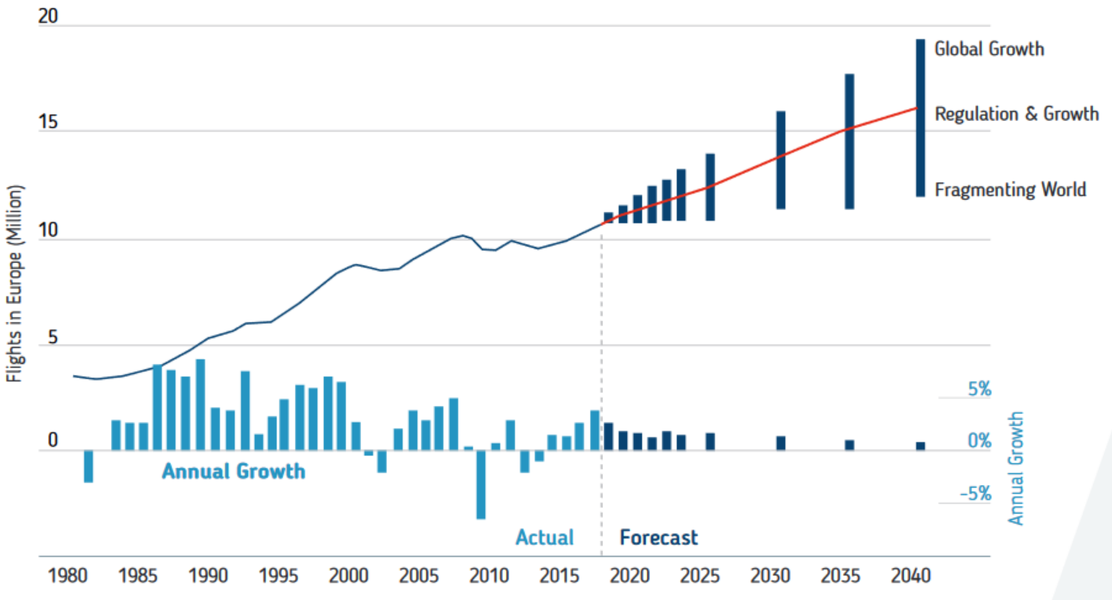

Figure 1: Overview of the future long-term traffic (IFR movements) [2]

In conclusion, air traffic demand will continue to increase over the next 20 years, in result,

[1] ICAO: specialized agency of the United Nations. It codifies the principles and techniques of international air navigation and supports the planning and development of international air transport to ensure safe and orderly growth.

[2] IFR stands for instrument flight rules – the set of rules that govern aircraft that fly in IMC, or instrument meteorological conditions. The pilot navigates only by reference to the instruments in the aircraft cockpit.

air traffic congestion will become an everyday problematic. In consequence, the European air transport system will have to turn into more agile to respond to the future challenges produced by the growing demand of air travel.

In the airline sector, there is a lack of Direct Operational Costs (DOCs)[3] management. There are three different types of airlines: commercial, cargo or charter where there are more than 5000 commercial airlines worldwide.

The cost management of commercial airlines is based on direct and indirect costs. The direct costs represent the 70% of the total airline costs; composed by: Navigation Taxes, Handling[4]Costs, Airport Taxes and Fuel Costs. The current DOC management it is known to be: paper based, ran with legacy technologies, labour-intensive, time consuming and manual data entry, etc. which results in overpayment for the airlines. In the future, as has been shown, the number of flights will increase and thus, DOCs management will be more challenging.

In order to control DOCs and reduce overpayment, Airplane Solutions has introduced a solution that provides a real-time and accurate data performance that reliably calculates costs; collecting data directly from the aircraft equipment during each flight and processing it to calculate airline DOCs using Blockchain technology.

This Bachelor Final Project is part of Airplane Solutions project to calculate en-route[5] charges, with high levels of accuracy, thanks to real data captured from the aircraft equipment. This will allow to compute the actual trajectory of each flight and calculate its charges.

Therefore, this research tries to visualize the difference in cost between the initial and actual flight plan, as the European En-Route Charges Systems bases on initial flight plan, as detailed in the following sections.

This project aims to demonstrate that the real total cost of the en-route phase is less than the cost based on the flight plan and therefore, prove savings in DOCs regarding the navigation costs for airline.

[3]DOCs: Expenses incurred directly in the operation of a particular aircraft. They are formed by Navigation, Handling, Airport and Fuel Costs.

[4]Service of assistance to airplanes on the ground that is to say, it is the set of terrestrial operations that allow that an airplane can be loaded and unloaded of passengers, merchandise and luggage.

[5]Part of the flight from the end of the take-off and initial climb phase to the commencement of the approach and landing phase

CHAPTER 1. PROJECT PROPOSAL

In this section, the principal objectives are described, as well as its scope.

1.1. Objectives

Hours before the flight takes off, the airline has to send the flight plan to the Eurocontrol Operations Centre in Brussels. There, a computer system checks that the flight plan is adapted to the rules of each aircraft on which the flight is to take place and approves it.

During the flight, this route can change due to various factors: meteorological conditions, airspace congestion, ATC[1] clearances, etc. The main objective of this project is to **analyse the total cost variation of the en-route phase between the real flight and the flight plan**. Nowadays, airlines are charged for initial flight. This study will result in a direct operational cost saving tool for airlines.

The second main objective is to compare the en-route charges obtained with this tool and the results extracted directly from NEST. So later, differences can be computed.

Moreover, this project will study costs variations regarding the countries and airlines. It also aims to help airlines perform more accurate flight plans in the future. In addition to that, it shows which routes are most affected by changes in trajectory by airline.

In addition to that, this project is created to improve air management systems in Europe.

Finally, this first tool version is an initial model for the calculation of the en-route charge for Airplane Solutions, as until now it has not been possible to compute it.

1.2. Scope

In this section, it is explained the limitations of this project. It describes what will be done and what will not be done in this first tool version.

This project does not integrate all European airspaces, the scope is reduced to the 17 Central European countries. Those countries were chosen due to the fact that the majority of the European Air Traffic is concentrated in their airspaces, and hold the European major airports.

In order to compute the simulations, flights from NEST Eurocontrol Application are being taken, because the Airplane Solutions prototype[2] is in process of installation and flight data is not available yet.

The analysis is based on flights performed by aircraft model A320 family, commercial twin-engine jet manufactured by Airbus. As a result, it is focused on airlines which integrate the A320 in their fleet. Nevertheless, in a future version, all aircraft model will be integrated.

Finally, this script is half-integrated in the Airplane Solutions Cloud Platform. Some variables are extracted from the Airplane Solutions data base but the calculation will be com-

[1] Air Traffic Controllers

[2] Airplane Solutions prototype is in process of device certification which capture data in real time from avionics

puted using Blockchain technology in a second project phase, as the other fees and taxes are being analysed.

CHAPTER 2. THEORETICAL BACKGROUND

2.1. State of Art

As exposed in the introduction section, air traffic is expected to grow, so the aim of this project is to help the development of air management systems and reduction of DOCs. Nowadays, institutions, research groups and experts are also working on new technologies to improve air traffic management. In this section, there is an analysis of current investigations, platforms and research groups which have the same objective as this project:

- **SkyWise Platform from Airbus:** [4]
  Airbus launched the Skywise aviation data platform at the 2017 Paris Air Show, in collaboration with Palantir Technologies – which is a pioneer in big-data integration and advanced analytics. Skywise aims to become the platform of reference used by all major aviation players to improve their operational performance and business results, as well as to support their own digital transformation. In time, Airbus aims to extend Skywise to become aerospace's data platform of reference.
  Skywise will reach new insights into the operation of aircraft, optimise maintenance, engineering and flight operations decision-making and reduce costs.

- **Single European Sky ATM Research (SESAR):** [5] SESAR is the mechanism which coordinates and concentrates all EU research and development (R&D) activities in ATM, pooling together a wealth experts to develop the new generation of ATM. SESAR's role is to define, develop and deploy whatever is needed to increase ATM performance and build Europe intelligent air transport system.
  SESAR's vision builds on the notion of trajectory-based operations and relies on the provision of Air Navigation Services (ANS) in support of the execution of the business or mission trajectory — meaning that aircraft can fly their preferred trajectories without being constrained by airspace configurations.

- **INDRA:** [6] Indra Avitech's mission is to support the creation of an integrated, safer, cleaner civil and military air traffic environment by providing the aviation industry with databases, tools and processes. They offer all the complete System Wide Information Management(SWIM)[1] compliant solutions to take air traffic management operations to the highest level of efficiency while following all the principles of SESAR and NextGen[2].

- **En route charges for ANSP revenue maximization**: [7] Université Libre de Bruxelles in collaboration with Università degli Studi di Trieste studied how to maximize the en-route charges revenues. These charges usually account for a significant part of the cost of a flight, and they can therefore influence the route choice: airlines may decide to fly longer routes to avoid countries with higher charges. They studied an algorithm to compute the optimal Unit Rate and apply it to a case study relying on real air traffic data and realistic flight cost figures through a Network Pricing Problem (NPP) formulation.

[1] SWIM is a global Air Traffic Management industry to harmonize the exchange of Aeronautical, Weather and Flight Information for all Airspace Users and Stakeholders.

[2] Next Generation Transportation System is a term for the continuing transformation of the National Airspace System (NAS) of the United States.

2.2. European En-Route Charges System

Since 1950s, the introduction of jet aircraft causes the increase of air traffic, thus making the complexity of the system grow. Accordingly, the European States required more advanced automated systems and the costs in providing air navigation services increased meaningfully. In addition, the States were aware that with the expansion of the civil aviation market they were providing services free of charge to airlines of other countries (for further information see reference [8]).

As a result, the European Organisation for the Safety of Air Navigation, commonly known as Eurocontrol, was founded in 1960. The seven EUROCONTROL Member States at the time decided to adopt a common policy for en-route charging, created a joint system for the establishment, billing and collection of en-route charges, and use EUROCONTROL for these purposes. As a result, the EUROCONTROL Central Route Charges Office (CRCO), was established in 1971 to operate a centralised system for the collection of route charges. Located at EUROCONTROL's headquarters in Brussels, CRCO recovers operational costs for the use of airspace and air traffic management (ATM) services provided to airspace users, which fly in the controlled airspace of EUROCONTROL's Member States.

The CRCO, a Directorate of the EUROCONTROL Agency, is functionally organised as follows:

- Front-office: “Billing, Customer Relations and Economics Division” (CRCO/BCE)
- Back-office: “Collection of charges, Accounting and Treasury Division” (CRCO/CAT)
- Support: “CRCO Information Services Unit” (CRCO/CIS)

2.2.1. State Members

The EUROCONTROL Route Charges System is a regional cost-recovery system - aligned with ICAO recommendations - that finance air navigation facilities and services and supports ATM developments. Contracting States must establish en route charging zones in the airspace. This implies that the en route services provided to airspace users are supplied by the Contracting State.

The billing and recovery of air navigation charges by EUROCONTROL is indispensable in ensuring that air navigation facilities and services are steadily financed and safely operated, paving the way for the future evolution of the pan-European Air Traffic Management (ATM) system in the context of the Single European Sky and the European ATM Master Plan (SESAR) (for further information see reference [14]).

The CRCO bills and collects, in addition to route charges, terminal charges on behalf of Member States and air navigation charges on behalf of non-Member States.

The following figure 2.1, extracted from the reference [9], shows the Member States classification by air traffic service provided by EUROCONTROL.

EUROCONTROL Route Charges (3 Information Circulars)	Member States of EUROCONTROL: Belgium, Luxembourg, Germany, France, United Kingdom, Netherlands, Ireland, Switzerland, Portugal, Austria, Spain, Greece, Turkey, Malta, Cyprus, Hungary, Norway, Denmark, Slovenia, Czech Republic, Sweden, Italy, Romania, Slovak Republic, Croatia, Bulgaria, Monaco, The former Yugoslav Republic of Macedonia, Moldova, Finland, Albania, Bosnia and Herzegovina, Serbia, Montenegro, Lithuania, Poland, Armenia, Latvia, Georgia and Estonia.
Terminal Charges (16 Information Circulars)	Denmark, Italy, Ireland, Moldova, Hungary, Croatia, Slovenia, Netherlands, Lithuania, Sweden, Greece, Bulgaria, Spain, Malta, Albania and The former Yugoslav Republic of Macedonia.
Air Navigation Charges (4 Information Circulars)	Belarus, Morocco, Uzbekistan and Egypt.
Communication Charges (1 Information Circular)	Ireland

Figure 2.1: Air Traffic Services and the corresponding Member States

Therefore, the charging zones of En-Route Charges are the Principal Member States of EUROCONTROL.

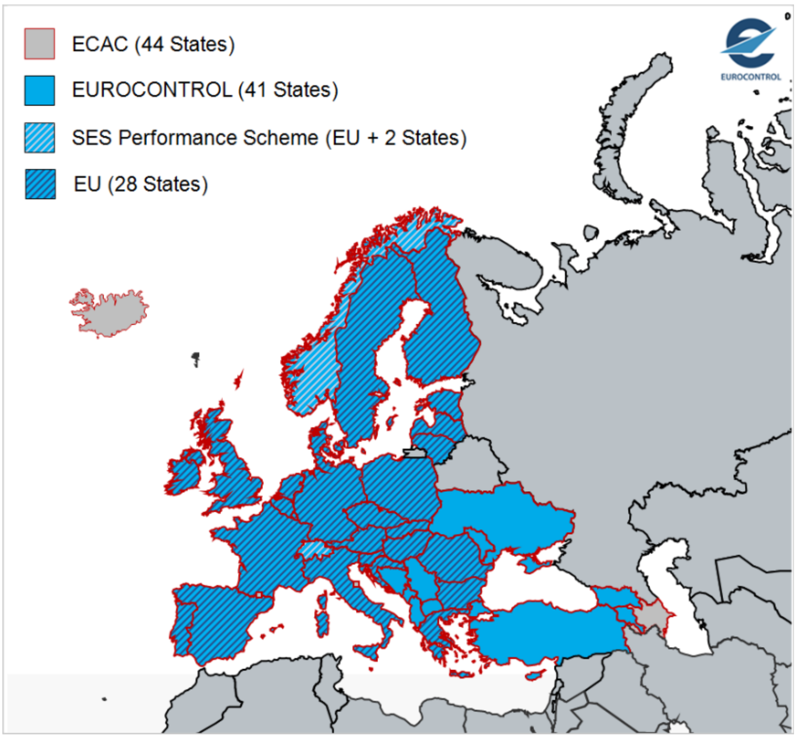

Figure 2.2: Classification of European Aeronautical Organizations

Previous figure 2.2 shows the principal European Aeronautical Organizations apart from

EUROCONTROL:

- **ECAC**: European Civil Aviation Conference. Intergovernmental organization which was established by International Civil Aviation Organization (ICAO) with 44 members: 28 EU, 31 of the 32 European Aviation Safety Agency member states, and all 41 EUROCONTROL member states. Its strategic priorities are safety, security and the environment. [10]
- **SES Performance Scheme**: Air Service Agreements between EU Member States and two others to establish Common EU rules in the aviation sector. [11]
- **EU**: European Union

2.2.2. Route Charges Principals

The basic principles of en-route charges system have been decided by the Member States of EUROCONTROL. The principles are the following:

- Regional: It covers EUROCONTROL Member States.
- Simple: There's a single charge per flight.
- Equitable: Route charges are paid by airspace users.
- Non-discriminatory: Computation of route charges with the same rules to all airspace users.
- Cost-related: Only air traffic management (ATM) costs are charges.
- Cost-effective: The objective is to collect at low cost and recover a high rate.

2.2.3. Flight Data Used For Route Charges Computation

The flight path used to calculate the distance flown in the airspace is extracted from the flight plan approved by the Network Manager (NM)[3]. EUROCONTROL operations area must **submit a flight plan** for every flight in European airspace. This flight plan, which contains detailed flight intentions, must be sent to the Network Manager Operations Centre.

2.2.3.1. Exempted Flights

There are some exemptions from the payment of route charges in the Contracting States and accordingly not implemented in en-route charge calculation. The exemptions are the following (for further information see reference [13]):

- Circular flights: For flights terminating at the aerodrome from which the aircraft has taken off and during which no intermediate landing was made.
- Flights by aircraft less than 2 tons.
- Search and Rescue flights.

[3]Network Manager: The Network Manager carries out air traffic management network functions for the European Commission, on behalf of EUROCONTROL which was nominated for this task. Working with its stakeholders, the Network Manager develops and runs the European ATM network (covering 43 countries), with the aim of meeting the Single European Sky's performance targets.[12]

- Military Flights.
- Training Flights.
- Flights under Visual Flights Rules (VFR).
- Humanitarian Flights.
- Customs and Police flights.
- Official mission flight (Monarch, Heads of Government and Government Ministers. . .).

2.2.4. Calculation of En-Route Charges

The three principal parameters to calculate the route charge are (for further information see reference [14]):

a Distance factor (DF)

b Weight Factor (WF)

c Unit Rate per State Member (UR)

$$\textbf{En-Route Charge} = \mathrm{DF} \times \mathrm{WF} \times \mathrm{UR} \tag{2.1}$$

These three parameters are defined in the following subsections.

2.2.4.1. Distance Factor

The distance factor is the division of the great circle distance by one hundred. The great circle distance is the number of kilometers between entry point of a charging zone (or departure airport) and exit point of a charging zone (or arrival airport) for each zone overflown.

The great circle distance is the shortest distance between two points on the surface of a sphere. Great circle routes require constantly changing headings. Long-distance air traffic uses great circle routes routinely, saving time and fuel (for further information see reference [15]).

The great circle distance (gc) is calculated according to the following formula (for further information see reference [16]):

$$\mathrm{gc} = 2 \times \mathrm{R} \times \arcsin\left(\sqrt{\sin^2\left(\frac{\delta-\delta'}{2}\right) + \cos\delta \times \cos\delta' \times \sin^2\left(\frac{\lambda-\lambda'}{2}\right)}\right) \tag{2.2}$$

where R = Radius of the Earth (6.367km)

δ = Initial latitude

δ'= Final latitude

λ = Initial longitude

λ' = Final longitude

The entry and exit points correspond to the last filed flight plan filed before departure and after any changes approved and/or Air Traffic Flow Management measures.

Figure 2.3 shows the difference between the great circle distance (direct route) with the flight trajectory through different charging zones:

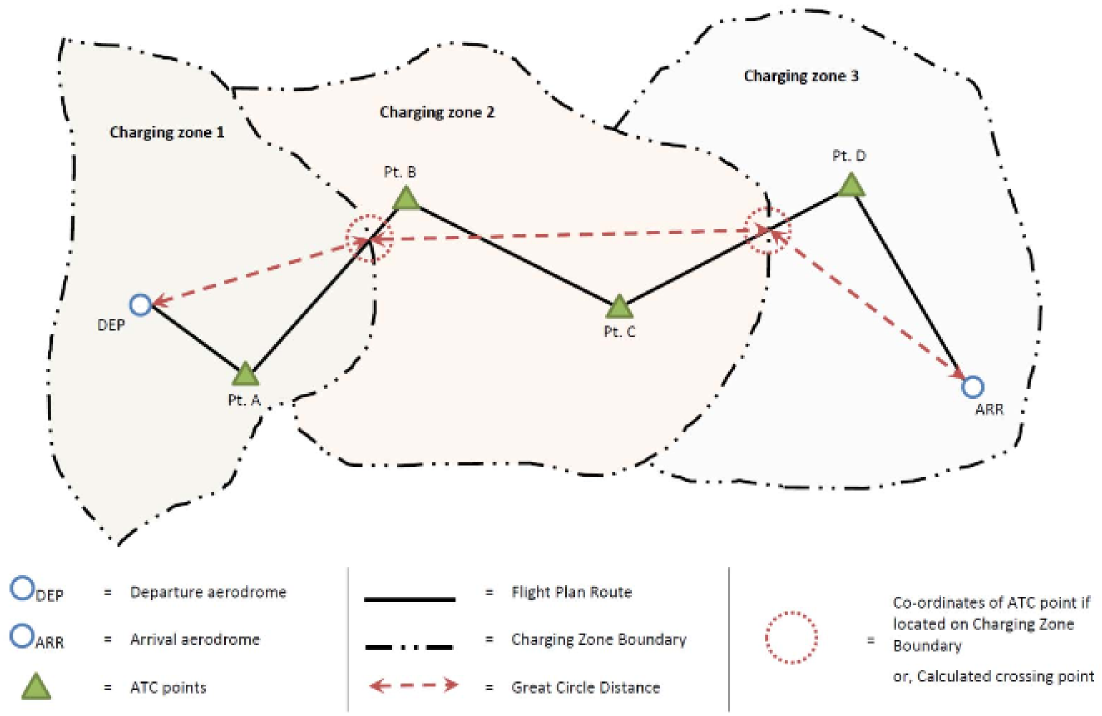

Figure 2.3: Representation of Great Circle Distance (in red) and the Flight Trajectory crossing some charging zones (in black)

As mentioned above, the distance factor is the great circle by one hundred, The DF is expressed by equation 2.3 (for further information see reference [14]).

$$\mathrm{DF} = \frac{\mathrm{gc}}{100} \tag{2.3}$$

where gc = great circle

2.2.4.2. Weight Factor

The weight factor is the division of the Maximum Take-Off Weight (MTOW)[4] of the aircraft (expressed to one decimal) by fifty and consecutively taking the square root of the result. (for further information see reference [14]). Therefore, the weight factor is defined by equation 2.4.

$$\mathrm{WF} = \sqrt{\frac{\mathrm{MTOW}}{50}} \tag{2.4}$$

[4]Maximum Take-Off Weight: value defined by the aircraft manufacturer. It is the maximum weight at which the aircraft is certified for take-off due to structural or other limits. [17]

The MTOW that the aircraft can declare must concorde with the certificate of airworthiness, the flight manual or any other equivalent official document. (for further information see reference [14])

2.2.4.3. Unit Rate of Charge

The unit rate of charge is the charge in euro applied by a charging zone to a flight operated by an aircraft of 50 metric tonnes (weight factor of 1.00) and for a distance factor of 1.00.

Basic unit rates are adjusted every month if the national currency of a Member State is not the euro. The monthly unit rate is recalculated by applying an exchange rate between the euro and the national currency (for further information see reference [18]).

Cost-bases and unit rates shall be calculated for each charging zone. There are two possible methods for the calculation of unit rates: one based on full cost recovery (hereinafter the “full cost recovery method”) and the other based on determined cost recovery (hereinafter the “determined costs method”).

a Full cost recovery method

Route charges shall be calculated for the year “n” based on the estimated costs and traffic for that year. An adjustment mechanism shall be applied to ensure that only the actual costs of the service are eventually recovered.

b Determined cost method

The determined costs shall be fixed prior to the beginning of each reference period as part of the performance plan for each calendar year during the reference period and in both real and nominal terms.

CHAPTER 3. DEVELOPMENT PHASE

3.1. Airspace Scenario

The base of this project is the European airspace division because the en-route charges are calculated in function of en-route airspace zone. En-route airspace is the volume of airspace outside terminal areas, where the climb, cruise and descent phases of flight take place and within which various types of air traffic services are provided (for further information see reference [21]).

Airspace is a portion of the Earth's atmosphere, both above ground and above water, regulated by each country, being defined depending on the movement of aircraft, the purpose of operations, and the level of safety required (for further information see reference [22]).

In order to be able to provide the various air traffic services, the world's airspace is divided into air navigation regions, 'EUR'. "NAT", "SAM", etc.), each covering several countries.

The boundaries of these regions do not coincide with the national divisions but are established according to the control requirements of each geographical area.

Vertically, we have 2 divisions of the airspace: the lower region or FIR[1] and the upper region or UIR[2]. The FIR extends from the ground up to flight level FL245 (24.500ft). The UIR goes from the upper level of the FIR FL[3]245 to FL460.

The en-route charges are defined by FIR and UIR. Each FIR/UIR is managed by a controlling authority that has responsibility for ensuring that air traffic services are provided to the aircraft flying within it.

The European FIR/UIR declared by ICAO are the following:

[1] Flight Information Regions: An airspace of defined dimensions within which flight information service and alerting service are provided

[2] Upper information Regions: FIR in upper airspace

[3] Flight Level

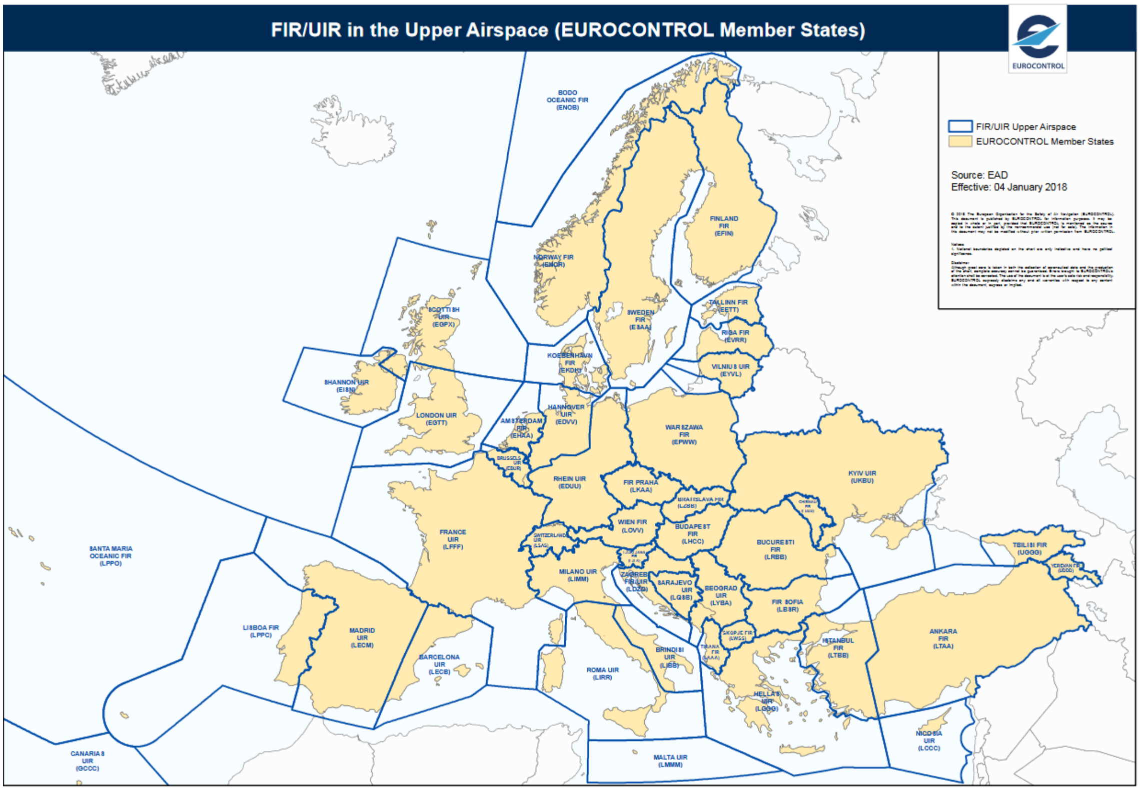

Figure 3.1: FIR Airspace division of EUROCONTROL Member States [23]

3.1.1. FIR/UIR borders definition

The objective of this section is to compute the coordinates of FIR borders for en-route charges. The coordinates of FIR are defined in the AIP[4] of each country.

For this first tool version, the airspace scenario scope is reduced to 17 countries, as explained in chapter 1. As the centre of air traffic congestion are the central European countries, they are chosen to perform this first tool version.

Table 3.1 defines the different airspace coordinates, from each country AIP, implemented in the tool:

[4] Aeronautical Information Publication

MEMBER STATE	AIRSPACE COORDINATES	AIP
Spain Continental	4500N 01300W;4500N 00800W;4420N 00400W; Spanish-French border; 4200N 00440E;3900N 00440E;3820N 00345E;3615N 00130W;3550N 00206W; 3550N 00723W; Portuguese-Spanish border;4200N 01000W;4300N 01300W; 4500N 01300W	AIP ESPAÑA ENR 2.1-1 WEF 08-NOV-18 [24]
Canarias Islands	3000N 02500W; 3000N 02000W; 3139N 01725W, from this point following a circle radius arc of 100 NM centered on 3304N 01621W up to 3130N 01545W; 3000N 01230W; 2740N 01310W; 2740N 01114W; 2120N 01400W; 2120N 01655W; 2047N 01704W; 1900N 01900W; 2400N 02500W; 3000N 02500W.	AIP ESPAÑA ENR 2.1-1 WEF 08-NOV-18[24]
Lisboa Continental	4300N 01300W - 4200N 01000W along border PORTUGAL_SPAIN - 3558N 00723W - 3558N 01200W - 3215N 01438W then a clockwise arc radius 100 NM centered on 330407N 0162130W -341504N 0174605W - 3630N 01500W - 4200N 01500W - 4300N 01300W	AIP PORTUGAL ENR 2.1.1 LISBOA FIR [25]
Santa Maria	4500N 04000W - 4500N 01300W - 4300 N01300W - 4200N 01500W - 3630N 01500W - 3415N 01746W - arc of circle with 100NM radius centered at PST NDB (anti clock-wise) 3140N 01725W - 3000N 02000W -3000N 02500W - 2400N 02500W -1700N 03730W - 2218N 04000W -to origin	AIP PORTUGAL ENR 2.1.5 SANTA MARIA OCEANIC FIR[25]
Malta	363000N 0113000E — 363000N 0190000E — 342000N 0233500E — 342000N 0113000E — 363000N 0113000E	AIP MALTA ENR 2.1 - 1 MALTA FIR[26]
Italy	433000N 0143000E 422606N 0160951E 422400N 0161600E 42250N 0162114E 413400N 0180000E 410800N 0185200E 385300N 0190000E 363000N 0190000E 363000N 0113000E 373000N 0113000E 390000N 0080000E 410000N 0080000E 412000N 0082000E 412000N 0094500E 431000N 0094500E 434700N 0073200E Italian northern geographical border till point 453700N 0134400E then line joining following points: 453800N 0133000E 451800N 0130000E 451000N 0130000E 443200N 0132000E 433000N 0143000E	AIP ITALIA ENR 2.1.1 -1 BRINDISI, ROMA AND MILAN FIR[27]
France	510700N 0020000E - 510521N 002334E - Frontière franco-belge - Frontière franco-luxembourgeoise - Frontière franco-allemande - Frontière franco-suisse - Frontière franco-italienne - 434703N 0073147E - 434700N 0073200E - 431000N 0094500E - 412000N 0094500E - 412000N 0082000E - 410000N 0080000E - 390000N 0080000E - 390000N 0044000E - 420000N 0044000E - 422700N 0031200E - Frontière franco-espagnole - Frontière hispano-andorrane - Frontière franco-espagnole - 432100N 0014700W - 433500N 0014700W - 442000N 0040000W - 450000N 0080000W - 485000N 0080000W - 500000N 0020000W - 500000N 0001500W - 504000N 0012800E - 510000N 0012800E - 510700N 0020000E	AIP FRANCE ENR 2.1.2 UIR FRANCE[28]
Switzerland	473400N 0074100E - 475300N 0085100E - 474700N 0085200E - 474730N 0091400E - 473930N 0091400E - National border with Germany, Austria (Liechtenstein included in FIR/UIR SWITZERLAND), Italy, France to 473400N 0074100E	AIP SWITZERLAND ENR 2.1 FIR SWITZERLAND[29]
Austria	465208.6161N 0160649.9210E - along State Boundary to - 463122.7488N 0134250.6758E - along State Boundary to - 465117.6926N 0102810.7570E - along State Boundary to - 473221.0120N 0093349.4028E - along State Boundary to - 484617.8329N 0135022.4354E - along State Boundary to - 483659.5406N 0165624.6784E - along State Boundary to - 480023.9623N 0170938.8034E - along State Boundary to - 465208.6161N 0160649.9210	AIP AUSTRIA ENR 2.1 FIR WIEN[30]
Germany	N550000 E0063000 – N550000 E0080000 – N550400 E0082000 – N550409 E0082331 – German-Danish border – N543840 E0110000 – N543610 E0111000 – N543315 E0112000 – N543000 E0113000 – N542750 E0114000 – N542645 E0115000 – N542700 E0120000 – N540800 E0111530 – German-Polish, German-Czech, German-Austrian and German-Swiss border – N473930 E0091400 – N474730 E0091400 – N474700 E0085200 – N475300 E0085100 – N473400 E0074100 – German-Swiss, German-French, German-Luxembourg and German-Belgian border	AIP GERMANY ENR 2.1 HANNOVER AND RHEIN UIR[31]
Belgium	510521N 0023244E - 510700N 0020000E - 513000N 0020000E - 512223N 0032147E - along the Belgian-Dutch border - 504515N 0060116E - along the Belgian-German border - 500748N 0060816E - along the German-Luxembourg border - 492810N 0062202E - along the French-Luxembourg border - 493247N 0054907E - along the Belgian-French border - 510521N 0023244E	AIP for BELGIUM ENR 2.1.1.1 BRUSSELS UIR[32]
United Kingdom	513000N 0020000E - 510700N 0020000E - 510000N 0012800E - 504000N 0012800E - 500000N 0001500W - 504500N 0012800E - 500000N 0020000W - 503800N 0011500E - 485000N 0080000W - 503600N 0011900E - 510000N 0080000W - 522000N 0053000W - 535500N 0053000W - 542500N 0081000W - 552000N 0065500W - 552500N 0072000W - 552000N 0081500W - 544500N 0090000W - 543400N 0100000W - 610000N 0100000W - 610000N 0000000E - 600000N 0000000E - 570000N 0050000E - 550000N 0050000E	UNITED KINGDOM AIP ENR 2.1 LONDON AND SCOTLAND UIR[33]
Ireland	5520N 00655W, 5425N 00810W, 5355N 00530W, 5220N 00530W, 5100N 00800W, 5100N 01500W, 5400N 01500W, 5434N 01000W, 5445N 00900W, 5520N 00815W,5525N 00720W, 5520N 00655W	IRELAND AIP ENR 2.1 SHANNON FIR[34]
Netherlands	550000.00N 0050000.00E; 550000.00N 0063000.00E; 534000.00N 0063000.00E; 533338.00N 0063624.00E; 533122.00N 0064020.00E; 533015.00N 0064430.00E; 532945.00N 0064859.00E; 532828.00N 0065149.00E; 532356.00N 0065658.00E; 532011.00N 0065937.00E; 531900.00N 0070130.00E; 531800.00N 0071130.00E; 531248.00N 0071301.00E; along Dutch-German border to 504515.44N 0060115.63E; along Dutch-Belgian border to 512222.76N 0032146.71E; 513000.00N 0020000.00E; to point of origin	AIP NETHERLANDS ENR 2.1 AMSTERDAM FLIGHT INFORMATION REGION[35]
Sweden	690336N 0203255E Swedish/Finnish border southward to - 653148N 0240824E - 644100N 0225500E - 633700N 0213000E - 632830N 0204000E - 631000N 0201000E - 614000N 0193000E - 610000N 0191905E - 601803N 0190756E - 601130N 0190512E - 593346N 0195859E -591524N 0203239E - 590000N 0210000E - 573410N 0200900E - 570000N 0195000E - 555100N 0173300E - 545500N 0155200E - 545500N 0150807E clockwise along an arc of 16.2 NM radius centered on 550404N 0144448E - 545500N 0142127E - 545500N 0125100E - 552012N 0123827E Swedish/Danish border northward to - 561253N 0122205E - 583000N 0103000E - 584540N 0103532E - 585332N 0103820E Swedish/Norwegian border northward to - 690336N 0203255E	AIP SWEDEN ENR 2.1 SWEDEN FIR[36]
Norway	630000N 0000000E - 630000N 0040000E - 640000N 0050053E - 650800N 0061600E - 653706N 0065026E - 654500N 0070000E - 661240N 0074228E - 671500N 0092521E - 700000N 0150000E - 702832N 0175917E - 712000N 0250000E - 712000N 0280000E - 710000N 0300000E - 702200N 0314300E - 700000N 0310800E - 694741N 0304904E - along the border between Norway and Russia to - 690307N 0285546E - along the border between Finland and Norway to - 690336N 0203255E - along the border between Norway and Sweden to - 585332N 0103820E - 584540N 0103532E - 583000N 0103000E - 580200N 0093130E - 570000N 0073000E -570000N 0060000E - 570000N 0055000E - 570000N 0050000E - 581640N 0030047E - 590504N 0013916E - 600000N 0000000E -(630000N 0000000E)	AIP NORWAY ENR 2.1-1 NORWAY FIR[37]

Table 3.1: Airspace coordinates summary

In some cases, the airspace borders correspond to country borders as it is shown in previous table (i.e. Spain-Portugal border). After searching in some open data formats, there are ".rds" text type with the country borders coordinates (for further information see reference [38]).

Using RStudio, an open source software, the border coordinates are extracted from these files and afterwards the airspace is modelled based on AIP specifications. The following figures are some examples studied:

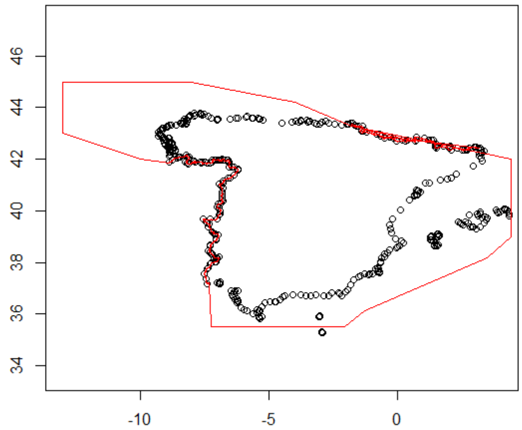

Figure 3.2: Representation of Spain FIR

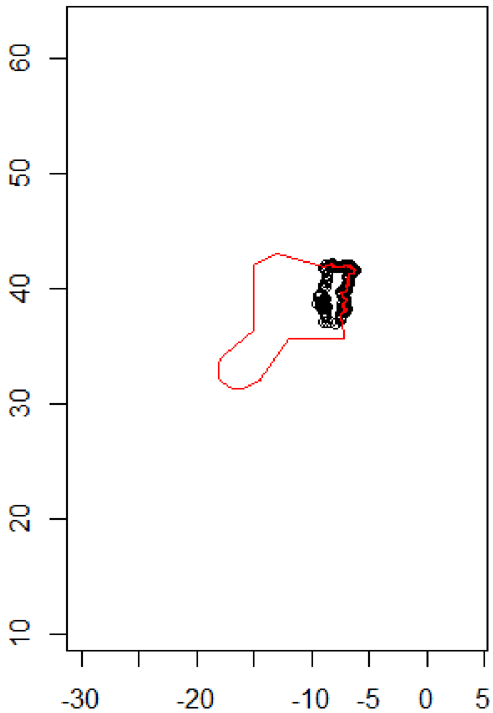

Figure 3.3: Representation of Portugal FIR

The airspace coordinates are saved in Excel file, processed by the script which is described in the next section 3.2. and plotted on a Software as a Service (SaaS) cloud computing platform, CARTODB. The final scenario is shown in figure 3.4.

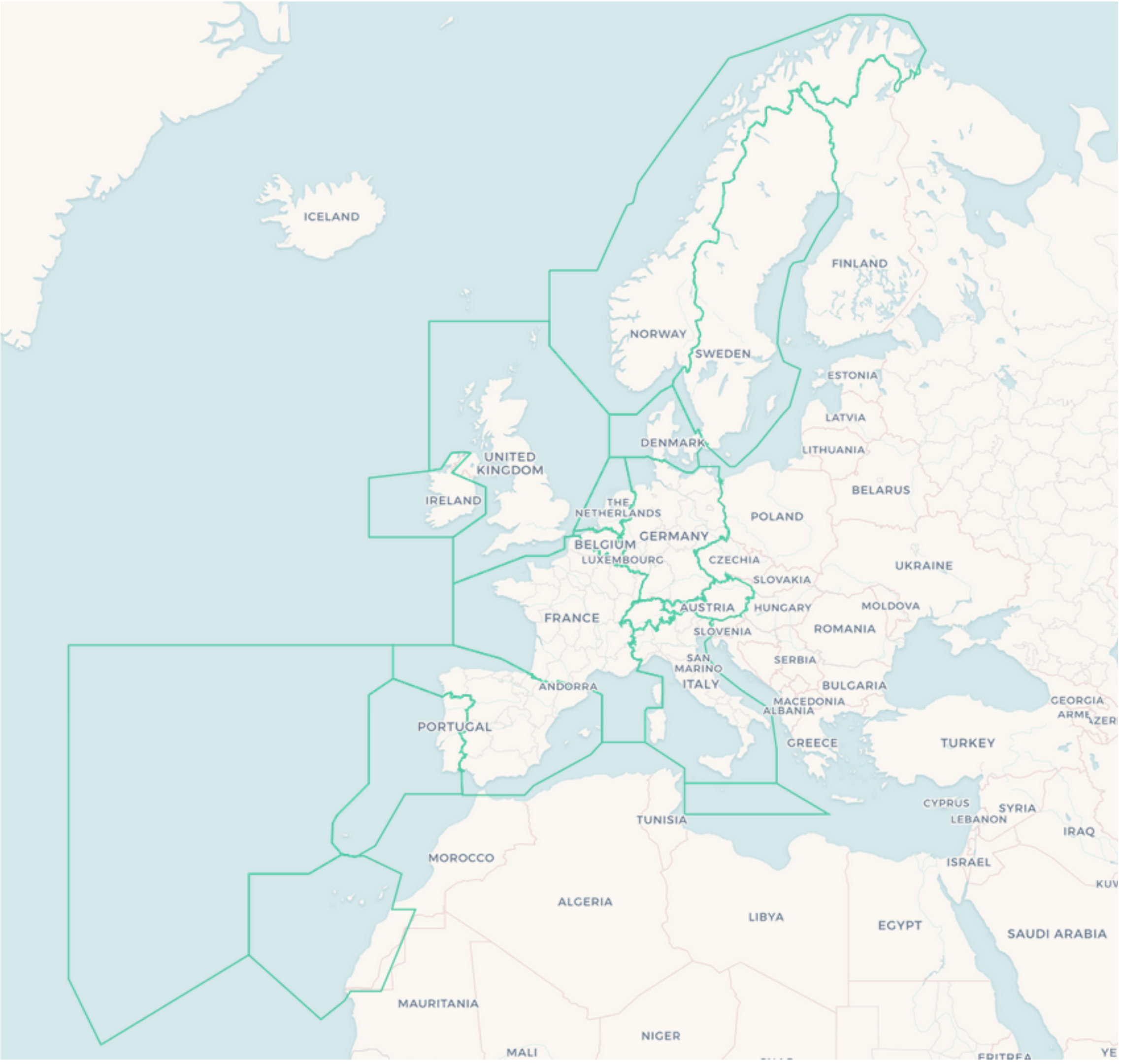

Figure 3.4: Tool Airspace Scenario
[39]

3.2. Script: En-Route Charges Calculation

The objective of this section is to describe the script, which it has been developed throughout the project in order to compute the en-route charges after processing the airspace coordinates.

n order to reduce the computational cost and to increment the performance of the script, the airspace coordinates are pre-processed before any calculations. This pre-processed coordinates are saved in a .pkl file so the program can directly consult with a pointer.

Figure 3.5 shows the structure of the script.

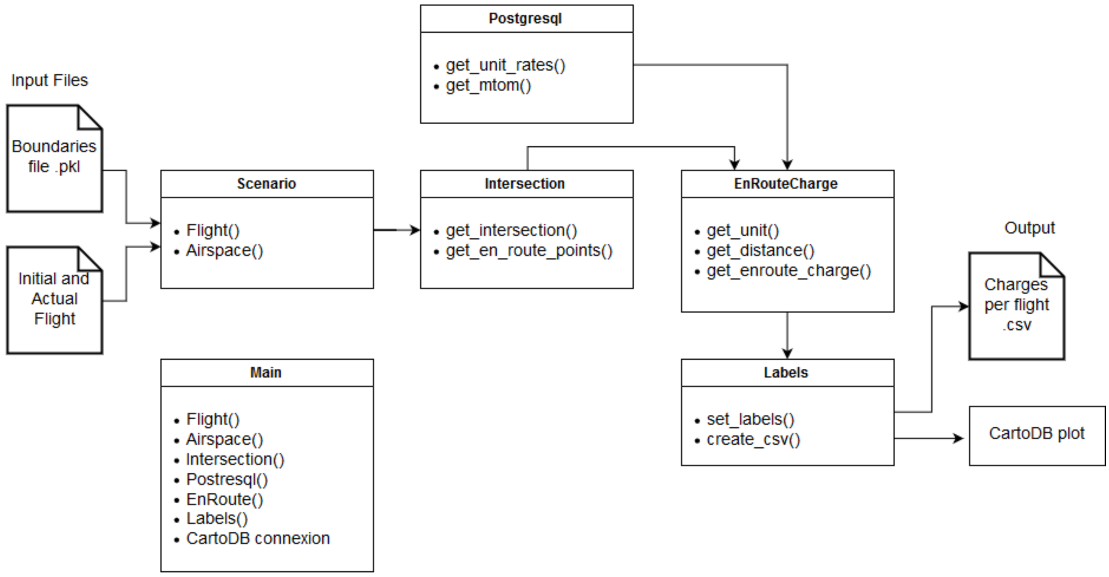

Figure 3.5: Class Diagram of En-Route Charges Calculation

The script En-Route Charges Calculation is described per function and classes in next sections.

3.2.1. Main Directories

The main directories of the code are:

- **Coordinates:** Directory from where the program reads the flights file.
- **SRC:** Directory from where the program reads the airspace .pkl files and the customised files to send to CartoDB Platform.
- **En-Route-Charges:** Directory where the code stores the output .csv files with results of en-route-charges per flight from Labels class.

3.2.2. Scenario

This module contains the two elementary classes for this project, to compute the working environment: Flight and Airspace.

The complex functions of **Airspace class** are:

- get_polys(): Identification of airspaces which specific flight crosses.
- get_countries(): Creation of polygons only from previously identified airspaces.

The complex function of **Flight class** is:

- get_flight_coordinates(): Input of flight information from Excel File and processed to compute the calculations.

3.2.3. Intersection

The principal objective of this class is to achieve the entry and exit crossing points for each flown airspace. There are two complex functions:

- get_intersection_points(): For each flight, calculates the intersection points between the different airspaces polygons.
- get_countries(): Returns a vector which identifies intersection points per country.

3.2.4. Postgresql

Unit rates are monthly adjusted and are published in Eurocontrol Portal (for further information see reference [18]), as explained in section 2.2.. Airplane Solutions has a data base (Shoganai Data Base) that automatically updates the new unit rates when they are posted monthly on the Eurocontrol website.

Also, there is a data base of aircraft models with their specifications, like MTOM. Both MTOM and unit rates are necessary for the calculation of en-route charge, as explained in section 2.2..

The functions of this class return this information from the data base. The functions are the following:

- get_unit_rates(): Returns the Eurocontrol Unit Rates from the data base.
- get_mtom(): Returns the MTOM from the data base.

3.2.5. EnRouteCharge

Once all the information required to compute the calculation is correct, it proceeds to calculate the costs.

The functions of this class are the following:

- get_unit(): Returns the unit rates necessary for an specific flight.
- get_distance(): Returns the great-circle distance between the two intersection points for each flown airspace.
- get_enroute_charge(): Returns the final en-route-charge applying the formula (2.1).

3.2.6. Labels

After the calculation process, *create_csv()* function creates an output file (.csv) with all the information necessary about en-route-charges. It describes the flight, the type of flight (initial (planned) or actual), country, cost per country and total cost. Figure 3.6 shows one output example of the function:

Flight	Category	Country	Cost
LEBL_EDDS	Initial	Spain Continental	154.36
LEBL_EDDS	Initial	France	399.32
LEBL_EDDS	Initial	Italy	23.29
LEBL_EDDS	Initial	Switzerland	265.16
LEBL_EDDS	Initial	Germany	89.67
LEBL_EDDS	Initial	Total	930
LEBL_EDDS	Actual	Spain Continental	154.36
LEBL_EDDS	Actual	France	361.3
LEBL_EDDS	Actual	Italy	111.16
LEBL_EDDS	Actual	Switzerland	255.65
LEBL_EDDS	Actual	Germany	108.79
LEBL_EDDS	Actual	Total	989

Figure 3.6: Example of output csv file describing the results

The function *set_labels()* customize the data to export them to CartoDB Platform, with cost per country, intersection points, accumulative cost and percentage of flight completed. There are some examples in the following section.

3.3. CARTO Platform

CARTO (for further information see reference [39]) is an open source tool that allows the storage and visualization of geospatial data on the web. With CARTO, it is possible to upload the geospatial data using a web form and then make it public or private. After it is uploaded, it can be visualised it in a table or on a map and apply map styles using CartoCSS.

As mentioned above, the script adjusts the data to export them to CARTO Platform The airspace scenario and the boundaries are part of the baseline scenario and they are integrated in map. This means that the script does not send the data of the borders, the platform already saves this data directly in map database.

The data exported is the following:

- Initial and Actual Flight coordinates
- Latitude and longitude of each intersection point
- En-Route Charge per airspace/country
- Total En-Route Charge per flight
- Percentage of flight completed

Export this data to CARTO Platform is possible thanks to carto-python APIs. Carto-python is an APi that allows the integration of CARTO into Python projects. Figure 3.7 and figure 3.8 are obtained with the CARTO platform.

Figure 3.7 shows an result example, where the initial and actual flight from Barcelona airport to Stuttgart airport is simulated.

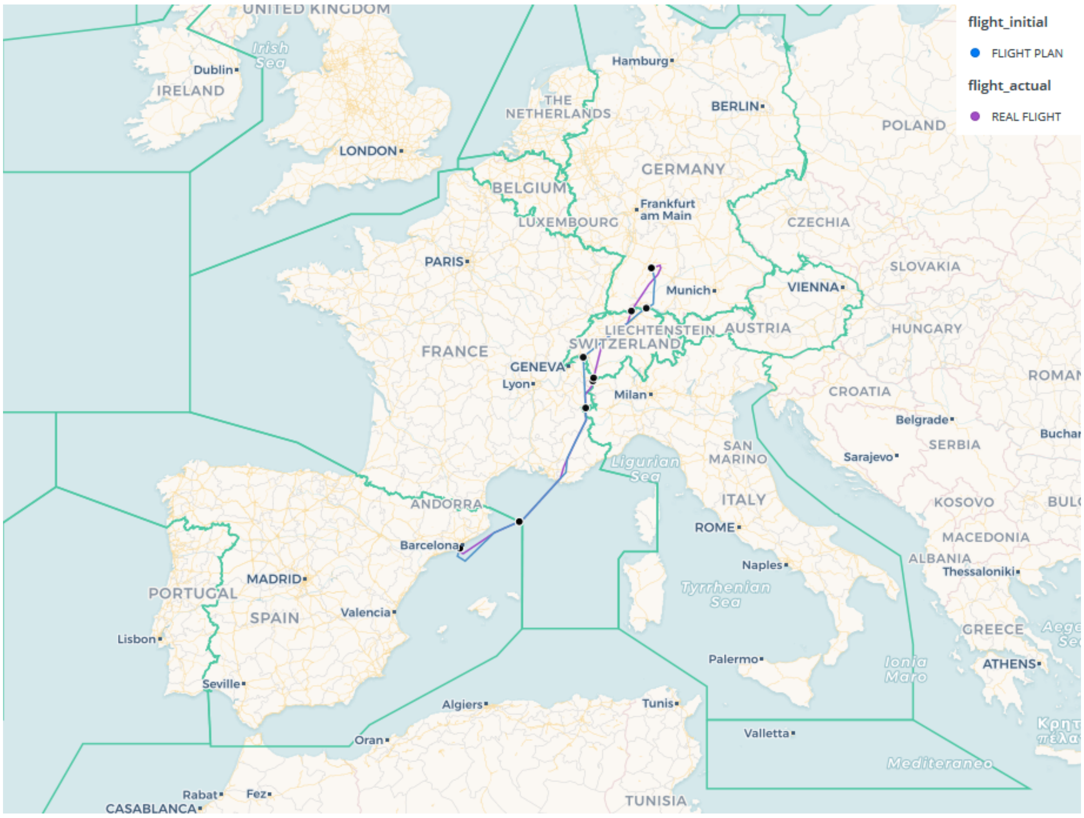

Figure 3.7: Example of a flight from Barcelona to Stuttgart

Each black point represents the intersection point with the specific airspace. It is a dynamic tool because at each point you can access information on the en-route charge, the cumulative cost of the flight and the percentage of the flight performed, for actual and initial flight. Therefore, it is possible to compare in an easy way the two types of flight. Figure 3.8 shows the information that the user can obtain throughout the tool.

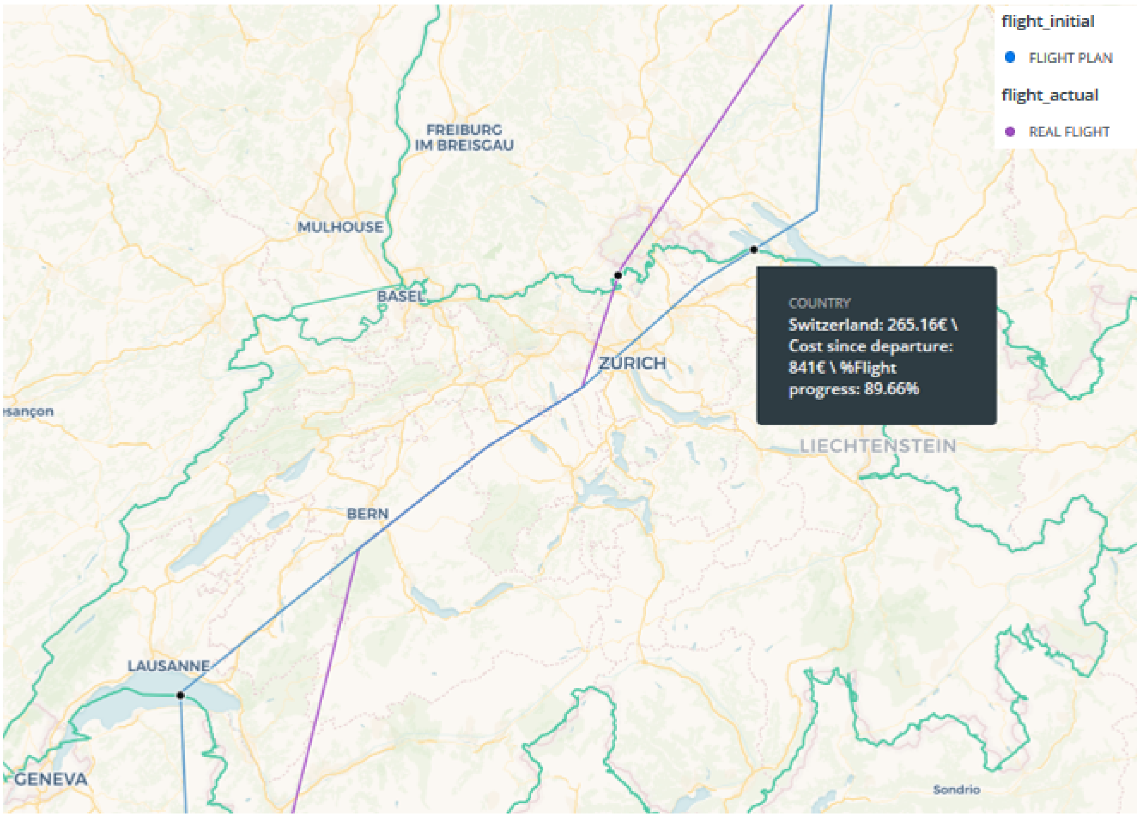

Figure 3.8: Example of a flight from Barcelona to Stuttgart

CHAPTER 4. ANALYSIS

The objective of this chapter is to define in detail the different simulations to prove the objectives of the project identified in chapter 1.

First, the methodology applied is defined in this final chapter. Also, the steps and conditions are justified to explain how the analysis will be performed. Finally, the results obtained with the tool are analysed.

4.1. Methodology

Before starting with the analysis process, it is important to define the methodology method that is applied. Mainly, there are two methodology types (for further information see reference [40]):

- **Qualitative methodology:** This type of methodology collects information that aims to describe an aspect, rather than measure it. They are made up of impressions, opinions and perspectives. A qualitative survey is less structured, it has the objective of deepening into the topic to obtain information about people's motivations, thoughts and attitudes. Although these data provide in-depth knowledge from the research questions, their results are more difficult to analyse.
- **Quantitative Methodology:** This second type of methodology allows to examine the data in a numerical way, especially in the field of Statistics. It is designed to collect concrete data, such as figures. These data are structured and statistical. They provide the support needed to reach general research conclusions. The main advantage of this methodology is that the results obtained will be extremely reliable. They present an honest picture of the conducted research without discrepancies and is also accurate.

The quantitative methodology is applied in this project as the main objective is to prove a cost variation between initial and actual flight. It is important to process flight data and export reliable results in a numerical way.

Another important point to consider to use the quantitative methodology type is that these results are processed and analysed in order to calculate the relative error of the samples. Also, the quantitative methodology allows to perform a structured study.

In conclusion, this type of methodology is the most accurate way to evaluate the project hypothesis. Once the methodology type is defined, the analysis process can begin.

4.2. Analysis Conditions

Before starting with the analysis the conditions and steps to analyse must be defined. In order to later on structure and classify the simulations, it is important to perform a previous study referencing to the conditions and steps applied.

In section 1.2., some limitations and conditions are defined. However, in this section these conditions are described in detail.

The airspace scenario is composed by 17 central European countries which are EURO-

CONTROL Member States. Also, these airspaces have the most congested air traffic routes and manage European major airports. The 17 central European countries are defined in table 3.1.

Consequently, the analysed flights only cross through these 17 airspaces. The flights that are not flying through these 17 airspace are not considered in the first tool version.

In subsection 2.2.3.1., it is explained that some flights are exempted from the payment of route charges in the Member States. This condition is applied to the En-Route Charge Calculation script.

Moreover, the domestic flights are also exempted in this first tool version.

Regarding the flight data, it is exported from the NEST Eurocontrol Application because nowadays the Airplane Solutions prototype is not completed to gather real data information during en-route phase. The study is focused on one day: 9th May 2019.

Commercial passenger jet type A320 is chosen to perform the simulations, therefore the flights that use this aircraft model are the ones analysed.

In consequence of the previous limitation, five major airlines, which have A320 in their fleet, have been chosen to realize the analysis. The five airlines are: British Airways (ICAO Code[1]: BAW), Lufthansa (DLH), Eurowings (EWG), EasyJet (EZY) and Vueling (VLG).

These limitations are the conditions applied in script En-Route Charge Calculation to perform the simulations and export all the results.

In order to compare the results obtained with the En-Route Charge Calculation script and the en-route charges imported from NEST, the relative error is computed by airlines and countries.

Before the global relative error is computed per flight, the relative error for each airspace that the flight is flying through must be computed. Equation 4.1 is used to calculate these relative error.

$$x = \frac{|N - E|}{N} \tag{4.1}$$

where

N = En-Route Charge from NEST

E = En-Route Charge from first version tool: En-Route Charge Calculation

After that, the error is averaged with the equation 4.2.

$$\bar{x} = \sum_{n=1}^{\infty} \frac{x_i}{n} \tag{4.2}$$

And for the error of the standard deviation of the mean value will be used the equation 4.3.

$$Error = \sqrt{\sum_{n=1}^{\infty} \frac{(x_i - \bar{x})^2}{n \times (n-1)}} \tag{4.3}$$

[1]The ICAO airline designator is a code assigned by the International Civil Aviation Organization (ICAO) to aircraft operating agencies, aeronautical authorities, and services related to international aviation, each of whom is allocated both a three-letter designator and a telephony designator.

Equation 4.2 and equation 4.3 are extracted from [41].

Finally, the results are processed and analysed through Tableau Desktop. Tableau is an efficient analysis platform for data. Interactive maps can be created automatically. It contains interactive dashboards and allows a visual analysis in real time (for further information see reference[42]).

Sections 4.3. and 4.4. contains the global results and the results per airline obtained.

4.3. Global Results

Before the analysis per airline, a first study from NEST data to prove the principal objective of this project is performed.

Figure 4.1 shows the cost variations per country that occur during 9 May 2019. In this case, all flights of all airlines are simulated, not excluding any aircraft models. However, those established by the European En-Route Charge System are still excluded.

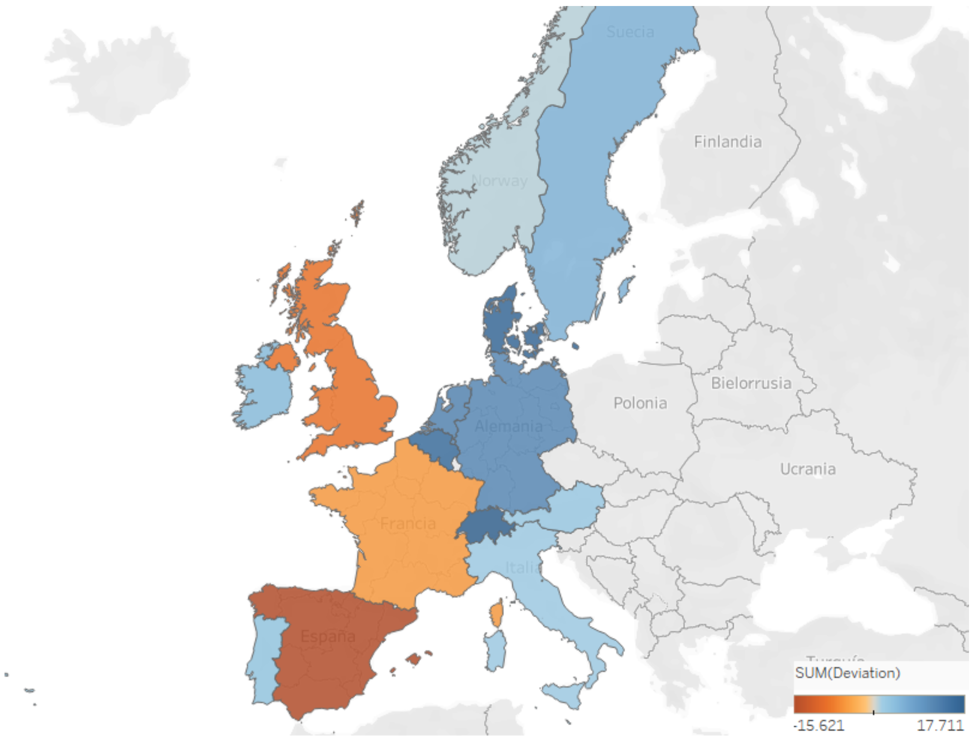

Figure 4.1: Representation of cost variation between planned and real flights on 9th May 2019

As map 4.1 shows, almost all the Eurocontrol Member States have an overpayment in Navigation Taxes (countries in blue colour). The positive difference (in blue) represents that the real routes are cheaper than the planned ones.

The main countries airlines that could save Navigation Taxes are Belgium, Switzerland, Germany and Denmark. It is comprehensible because these countries harbour the main airports of Europe and those airspaces manage a lot of air traffic. The controllers managing these airspaces tend to give more direct directions than the flight plan and therefore, there are more changes of route.

Furthermore, it must be considered the fact that on 9 May 2019, France restricted flights by 30% due to the general services strike. The airspaces around France are significantly affected both positively and negatively. Spain airspace, for example, has a negative difference, it means that the real trajectories are less efficient than the planned ones. One possible reason is to try to avoid the congestion because of the restriction in French airspace.

Figure 4.2 shows the cost variation between real and planned flights of a day without a lot of delays or congestion, to prove the cost variation in a typical day.

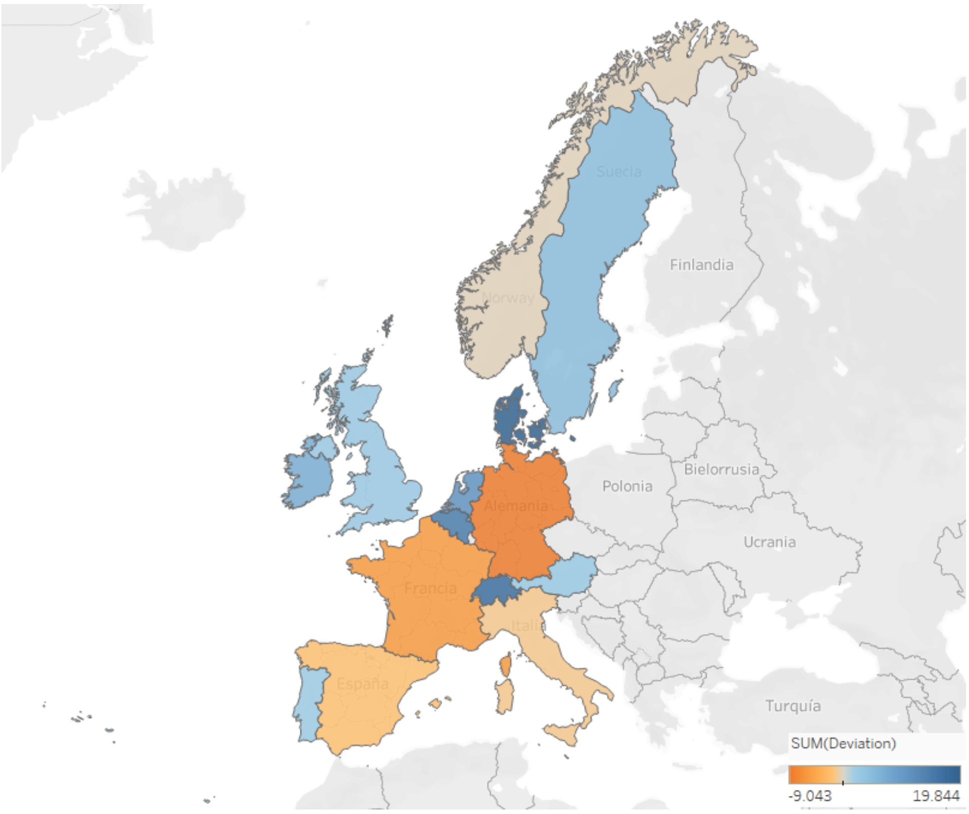

Figure 4.2: Representation of cost variation between planned and real flights on 8th May 2019

As it shows map 4.2, there is no impact as negative as in the previous map 4.1. The ranges of cost variation between initial and real flights are not as wide as the previous ones.

In the case of United Kingdom and Spain, it can be clearly seen that they do not have such a big difference. In the case of Germany, in the last analysis, this country has a positive difference but on 8th May 2019, the planned flights costs less than real flights.

In conclusion, the main countries that have more cost difference, have it both in a more congested day and less. Obviously, the most congested and space-restricted days have a greater difference. But as a conclusion, in trend, these countries are the most affected.

4.4. Results by Airlines

In this part, it is exhaustively explained the cost variation per airline and what impact this fact has on each of them and which routes are the most affected. Moreover, the error is analysed in comparison to the NEST application per country. Finally, it is shown an analysis per State Member. The results are based on 9th of May.

4.4.1. British Airways

British Airways is the national airline of the United Kingdom. The principal airport operational base, its main hub[2], is the London Heathrow Airport.

British Airways Global Analysis

Firstly, a first global analysis of the cost variation is carried out to show if this airline would benefit from the application. Furthermore, it will be able to analyse which countries are most affected and why.

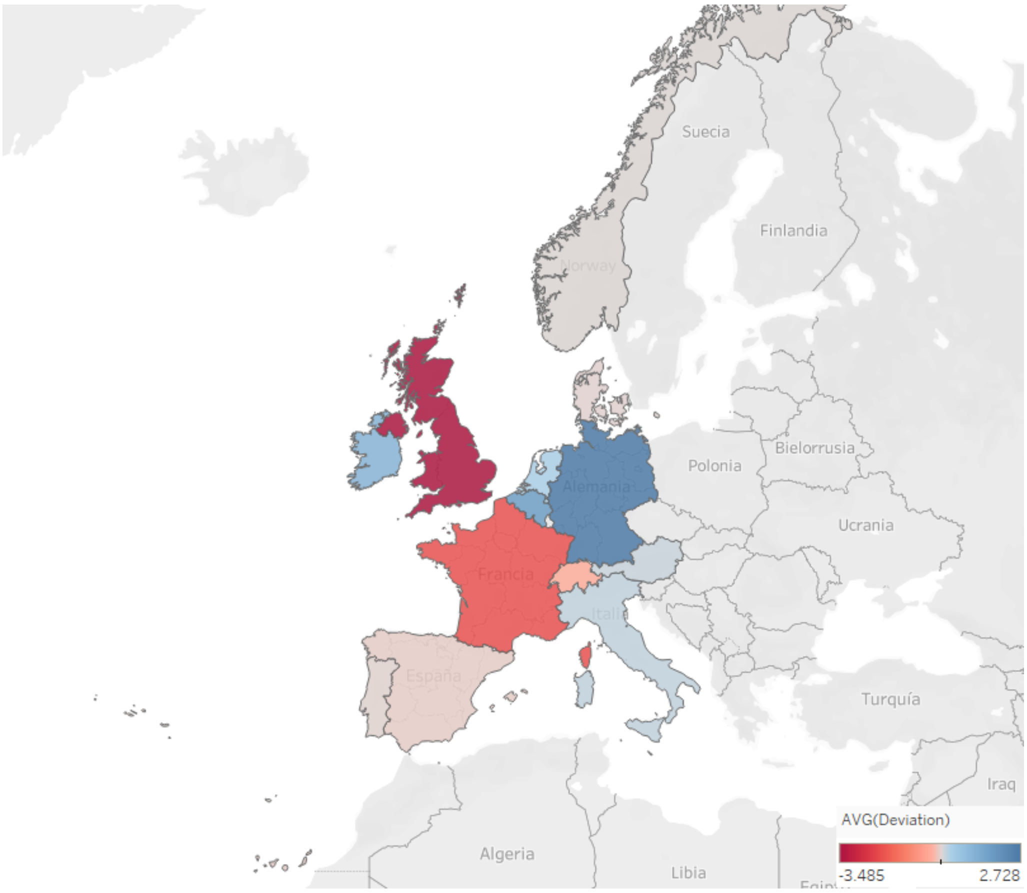

Figure 4.3: BAW Representation of cost variation between planned and real flights on 9th May 2019

As said before, UK is the country where its operations are focused. In map 4.3, the results for UK are negative, implying that real flights costs more than planned flights, because of the number of airline flights operating in this country. Furthermore, on 9th of May the France airspace was restricted and that limited their operations and we can also see a negative impact in France.

[2]Airports used by one or more airlines to concentrate passenger traffic and flight operations at a given airport. They serve as transfer (or stop-over) points to get passengers to their final destination.

In contrast, Germany, Belgium and The Netherlands show a positive difference, there are Navigation taxes savings. Other countries like Spain or Italy have neutral results. It probably means that the operation in these countries is not as important as in others.

Figure 4.4 shows these cost differences in numbers.

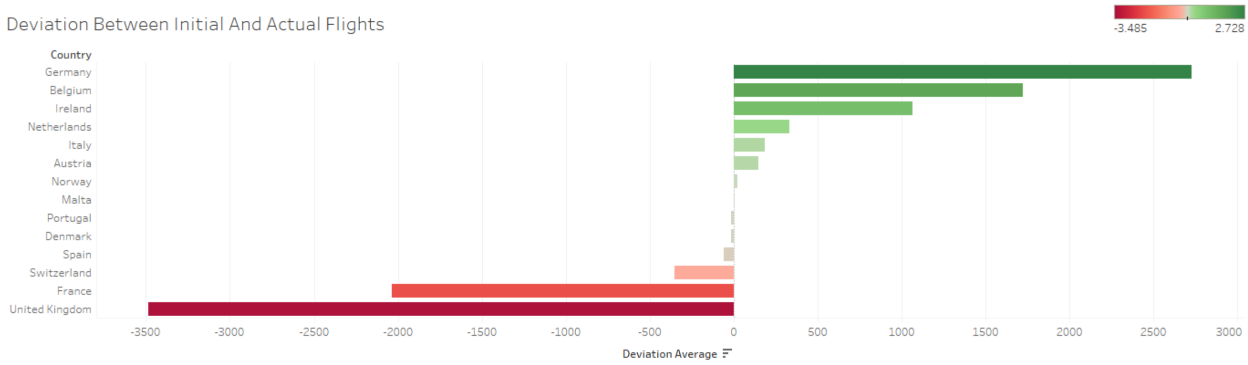

Figure 4.4: BAW Cost variation between planned and real flights per country on 9th May 2019

The range reaches a negative maximum. Therefore, the benefit lies more in the cost of initial flights. That does not mean, that in countries like Germany have an important positive maximum that must be considered.

Ireland is another country to consider. Because of its greater number of connections with the UK than with any other country, and therefore the airline's operations between these two countries, Ireland has a significant positive impact.

In conclusion, the results of this British Airlines are quite equal, the average of the countries have much positive difference and the rest negative, approximately.

Relative error compared to NEST application

In this next part, 23 flights with different origins and destinations are analysed, limited to the airspace scenario of this first version and countries in which the airline operate. The error is shown compared to the NEST application and the analysis of the most affected routes.

To compare the error between NEST application and En-Route Charges Calculation script, the formulas used are described in section 4.2..

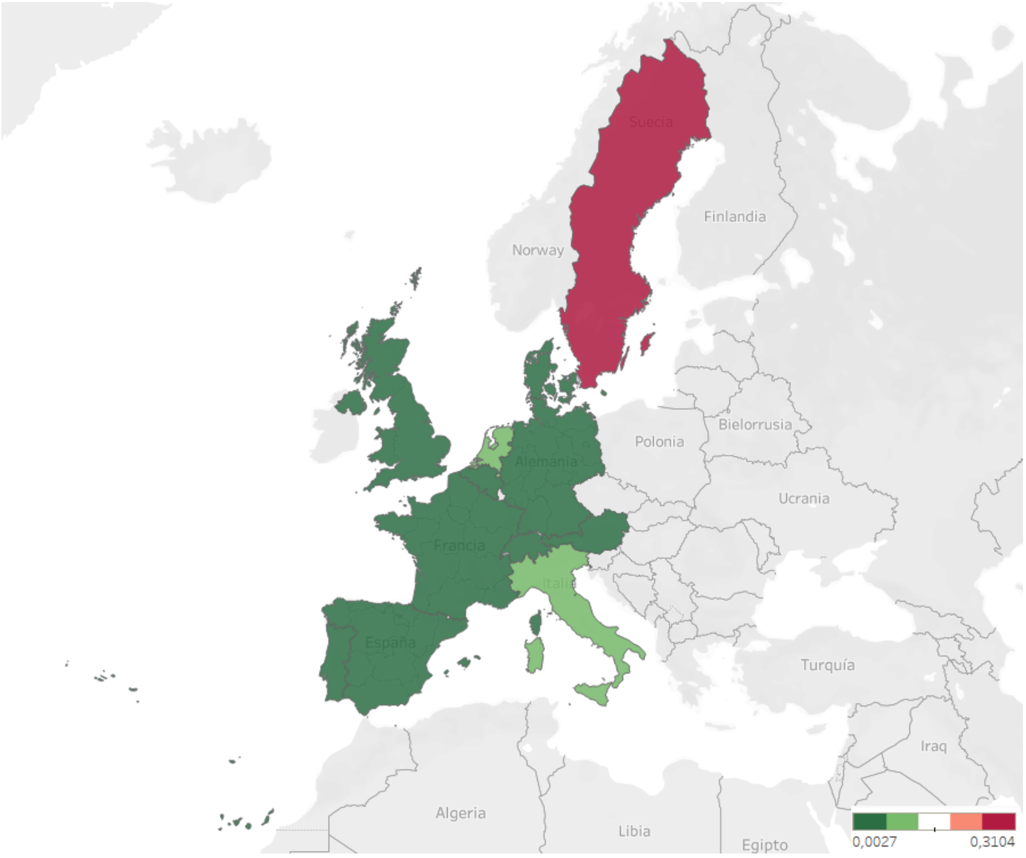

Figure 4.5: BAW Representation of Relative Error with NEST

Map 4.5 shows the countries with a greater error than the other countries where this airline operates. In general conditions, the relative error is small enough to validate the results except for the country Sweden.

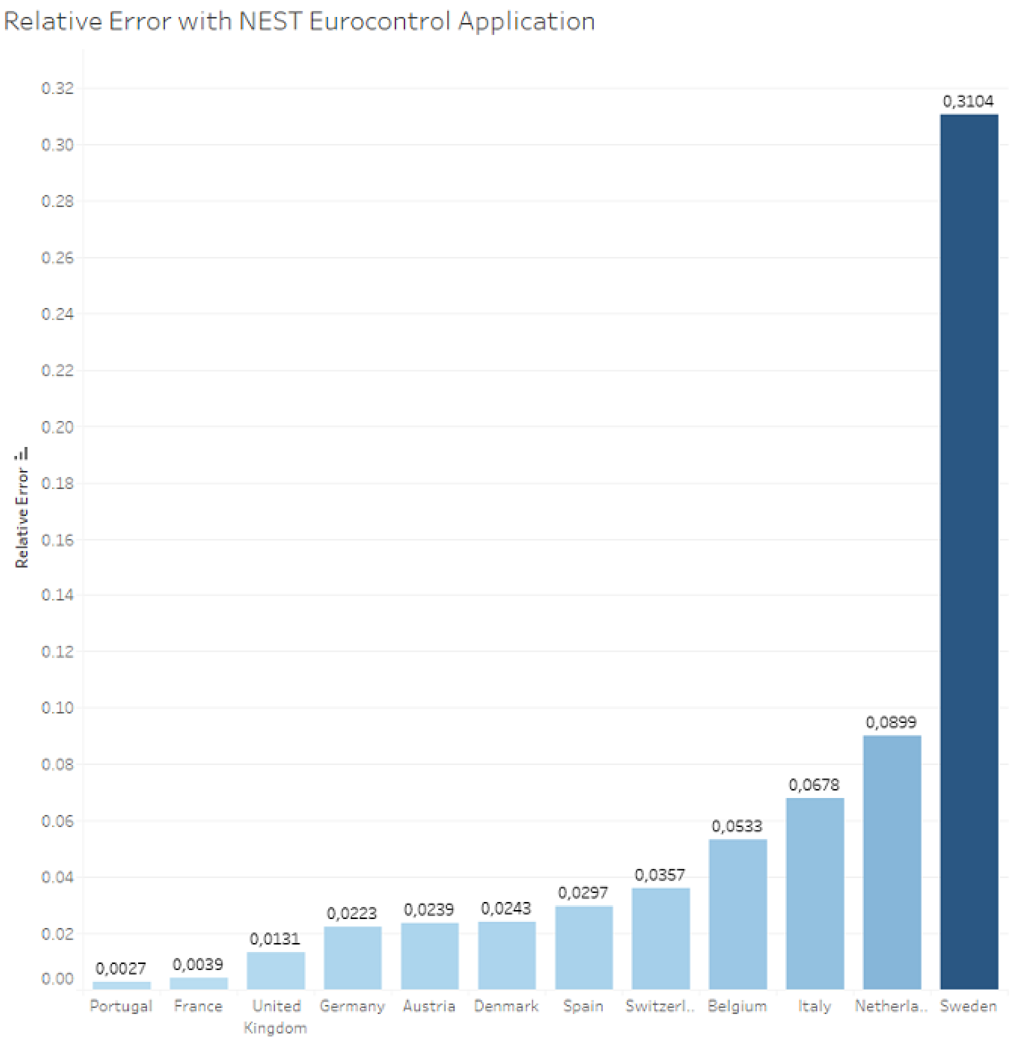

Figure 4.6: Relative Error per country of BAW airline

With graphic 4.6, it can be analysed the error per country in detail. As previously mentioned, Sweden is the country with the worst results. The reason is the few flights that fly over this country and that therefore not so many samples are found. The other countries, as it is described above, they do not have a major error.

Analysis of British Airways routes

Finally, in this section, the different routes are analysed in order to study which one have a more cost variation between the planned and real route. Then, the reasons are justified, and some conclusions can be reached.

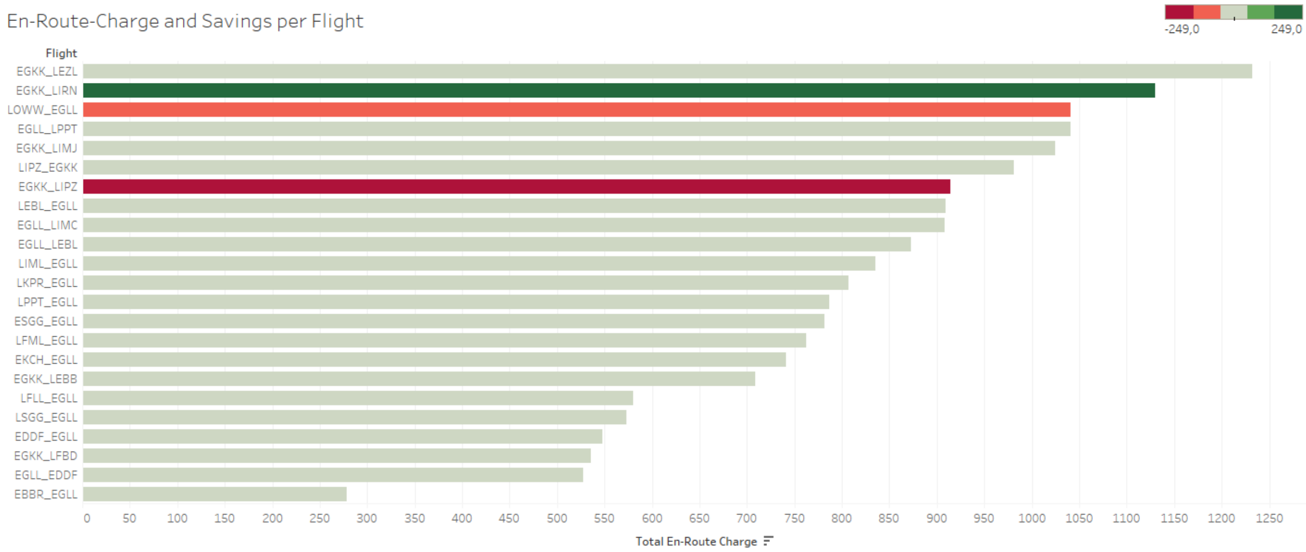

Figure 4.7: En-Route Charges per BAW Flights

In graph 4.7, the total en-route charge of BAW flights is represented, from the most expensive to the most economical. As it can be proved, the longer the route, which crosses more airspaces, the more expensive it gets. However, in this analysis, the cost variation of flights is performed.

In this case, almost all routes have a slight cost variation (represented in grey). The flight from United Kingdom to Italy is the most expensive. It may be because this route crosses the principal congested airspaces and their initial route can be modified. Also, these airspaces have high unit rate values. Although, the other affected routes, which have a negative difference and therefore the cost of the initial flight plan is more economical, also cross through the same airspaces. This means that whether they are a positive or negative cost difference, the routes through these areas of Europe will be directly affected and changed.

4.4.2. Lufthansa

Lufthansa is a German airline based in Cologne, considered Europe's largest airline since 2009 (for further information see reference [43]). Its base airport is Frankfurt with Munich International Airport as the second hub of operations.

Lufthansa Global Analysis

Firstly, a global assessment of the airline is shown, followed by a country-by-country analysis of its results.

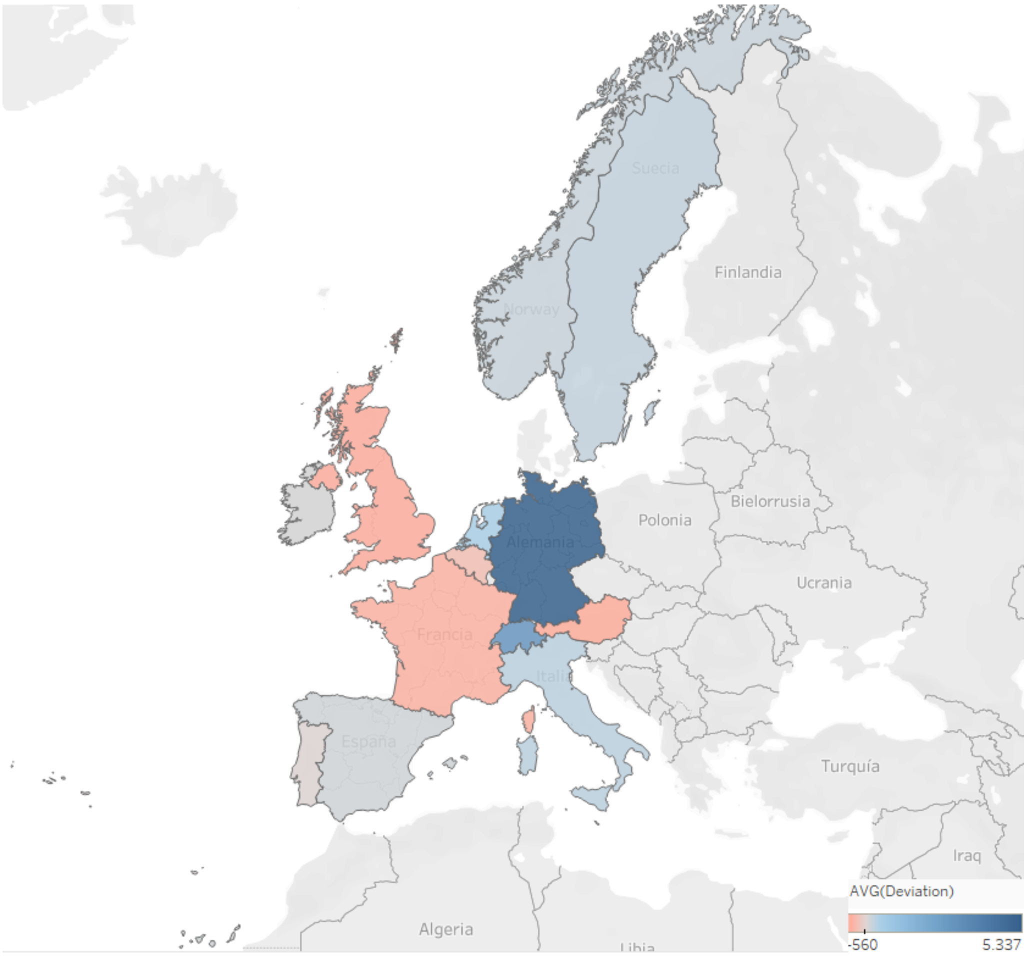

Figure 4.8: DLH Representation of cost variation between planned and real flights on 9th May 2019

In this case, few countries show a significant difference in costs. However, the range of variations shows that there is more positive than negative variation. The negative maximum is very small compared to the positive maximum. This means that globally this airline could significantly reduce its navigation costs.

The country with the most cost reduction would be Germany, its principal operational country. Furthermore, Spain, Sweden, Norway, etc. has a neutral impact.

France and United Kingdom has a have a small negative impact as the difference would be very small. But they show and justify that in those two countries, since an airspace restriction took place in France, more flights from those areas were affected.

Figure 4.9 shows these cost differences in numbers.

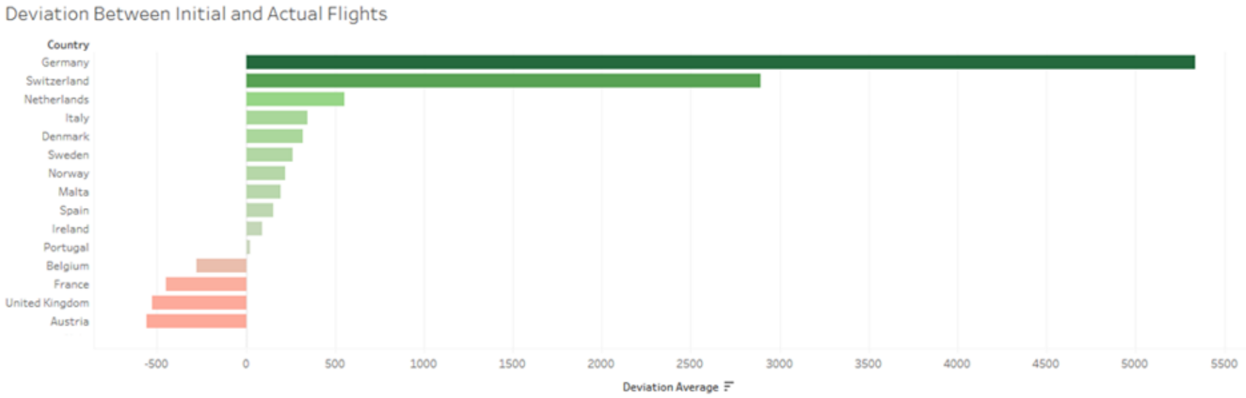

Figure 4.9: DLH Cost variation between planned and real flights per country on 9th May 2019

Graphic 4.9 shows that there is a lower number of countries with negative than positive impact, in contrast to the BAW airline. But Germany also has a more positive impact than other countries.

As mentioned in section 4.3., Germany and Belgium are two countries where Air Traffic Controllers allow more direct routes and therefore, changing their flight plan, in the time they cross their respective airspaces to reduce congestion.

In conclusion, the results of this Lufthansa are more positive, most of the countries have much beneficial cost variation.

Relative error compared to NEST application

In this section, 18 flights with different origins and destinations are analysed, limited to the airspace scenario of this first version and countries in which the airline operate. The error is shown compared to the NEST application and the analysis of the most affected routes.

To compare the error between NEST application and En-Route Charges Calculation script, the equations described in section 4.2. are used.

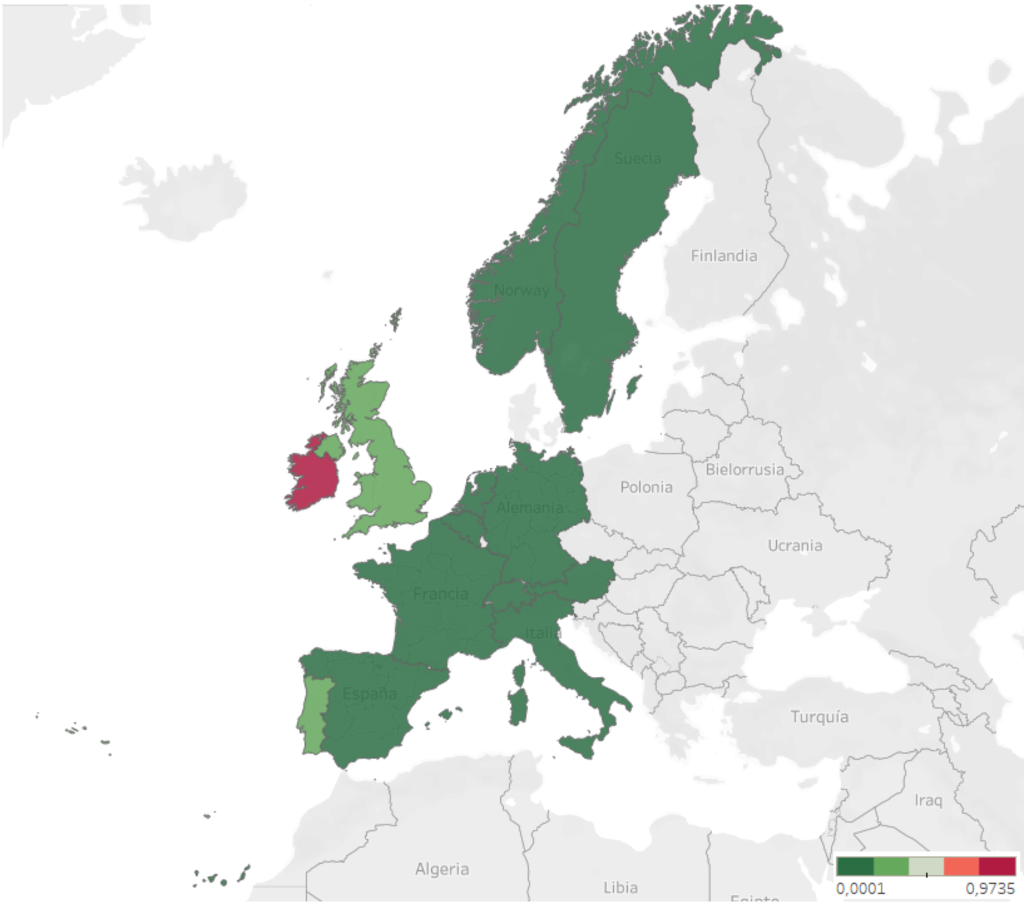

Figure 4.10: DLH Representation of Relative Error with NEST

Map 4.10 shows the countries with a greater error than others where this airline operates. In general conditions, the relative error is small enough to validate the results except for the country Ireland.

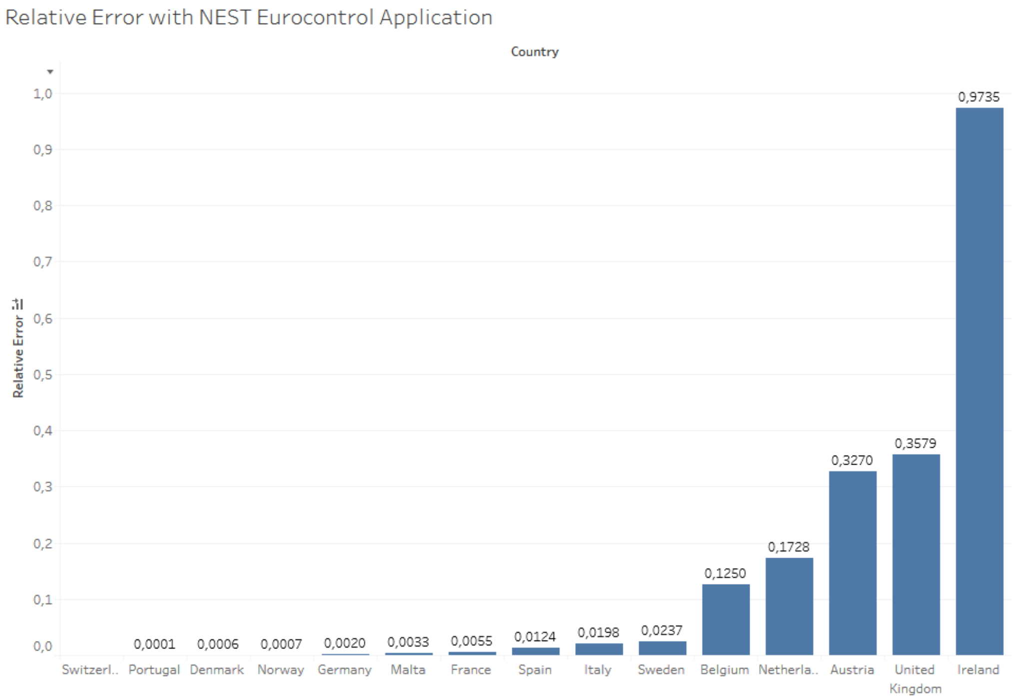

Figure 4.11: Relative Error per country of DLH airline

With graphic 4.11, it can be analysed the error per country in detail. As previously mentioned, Ireland is the country with the worst results. The reason is the few flights that fly over this country and that therefore not so many samples are found. With a larger number of samples, the error can be computed more accurately. The other countries, as it is described above, they do not have a major error.

It should be noted that these results are more accurate than the previous ones. But most of the countries with slight errors such as Portugal, Germany, Spain, France, etc. have the same trend with both airlines already analysed.

Analysis of Lufthansa routes

Finally, in this section, the different routes are analysed in order to study which one have a more cost variation between the planned and real route. Then, the reasons are justified and some conclusions can be reached.

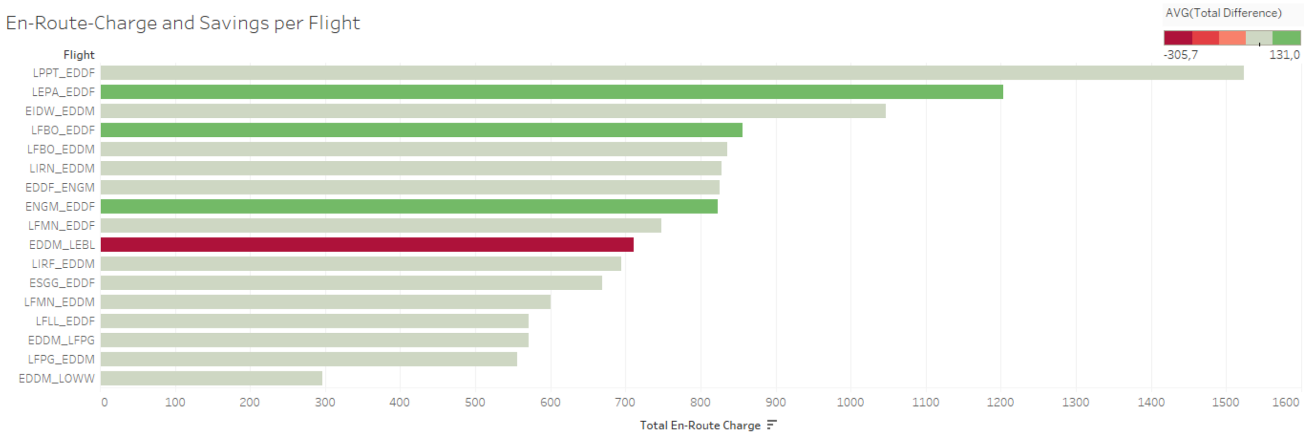

Figure 4.12: En-Route Charges per DLH Flights

Graphic 4.12 shows the total en-route charge of DLH flights. As mentioned above, the longest routes, which crosses more airspaces, are more expensive. In this case, almost all the flights has a slight cost variation (represented in grey). But, it is important to study the cost difference.

First, the range of cost difference is larger on the negative axis (-305.7€) than on the positive axis (131€). But it doesn't imply that the initial routes are cheaper than real routes. The number of flights with a positive cost difference is higher than with a negative cost difference.

Once again, routes with a significant cost deviation cross through the European central airspaces and through Germany. In all the results, Germany is being a focus of change of trajectories resulting in cost difference.

Finally, an important point to mention is that flights with higher en-route charges have significant cost differences, which could be minimized.

4.4.3. Eurowings

Eurowings is an airline based in Düsseldorf, Germany. It is a low-cost airline dependent on Lufthansa, which operates scheduled flights to destinations in Europe, Africa, America and Asia.

The company is made up of different airlines that operate all (or part) of their flights under the common name of "Eurowings". These airlines are currently: Eurowings, Eurowings Europe, Germanwings and SunExpress Deutschland.

Eurowings Global Analysis

Firstly, a first global analysis of the cost variation is carried out to show if this airline would benefit from the application. It will be performed an analysis by country.

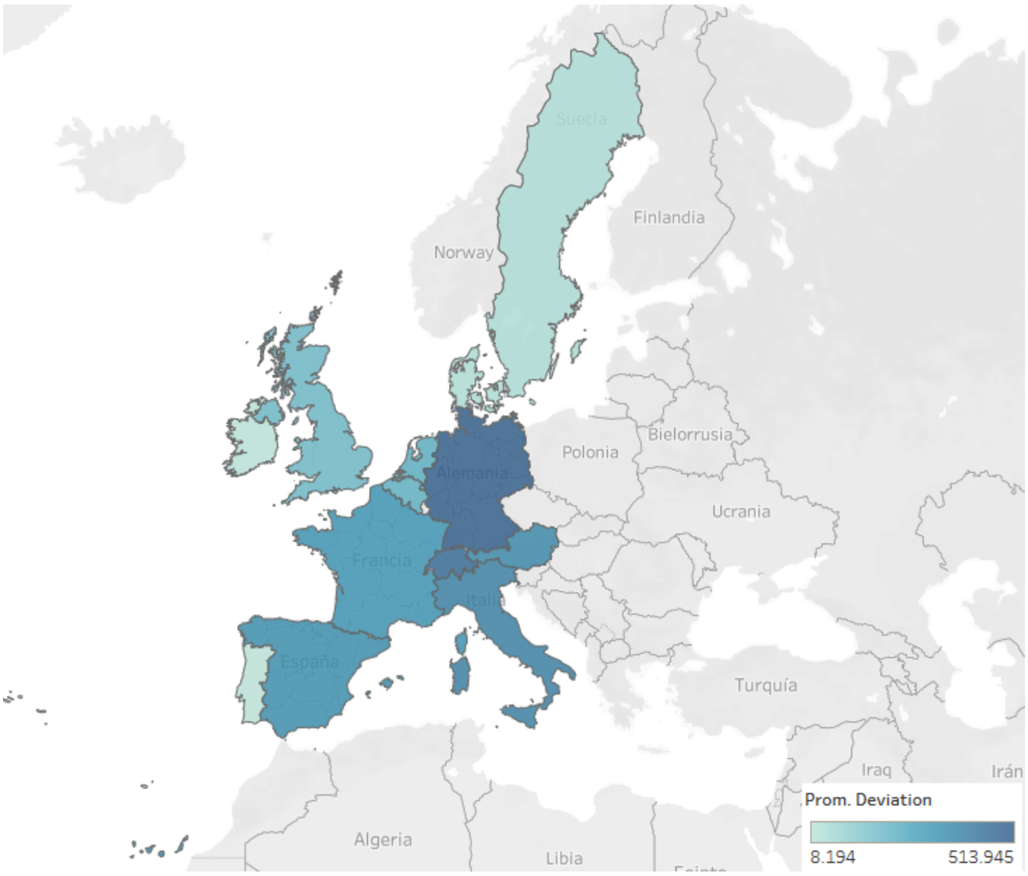

Figure 4.13: EWG Representation of cost variation between planned and real flights on 9th May 2019

In this case, all the different airspaces where Eurowings operate show positive results in the fact that cost variation is beneficial in the implementation of actual flight plans.

Furthermore, the ranges show considerable values to consider. In general terms, this airline can benefit from this reduction in the costs of Navigation Taxes.

The most outstanding countries are Germany, Italy and Switzerland. This airline mainly operate in these countries, which makes the variation greater. For example, countries such as the United Kingdom or Belgium, that normally have lot of air traffic, for Eurowings this countries are not very critical as they do not have a big difference.

Figure 4.14 shows these cost differences in numbers.

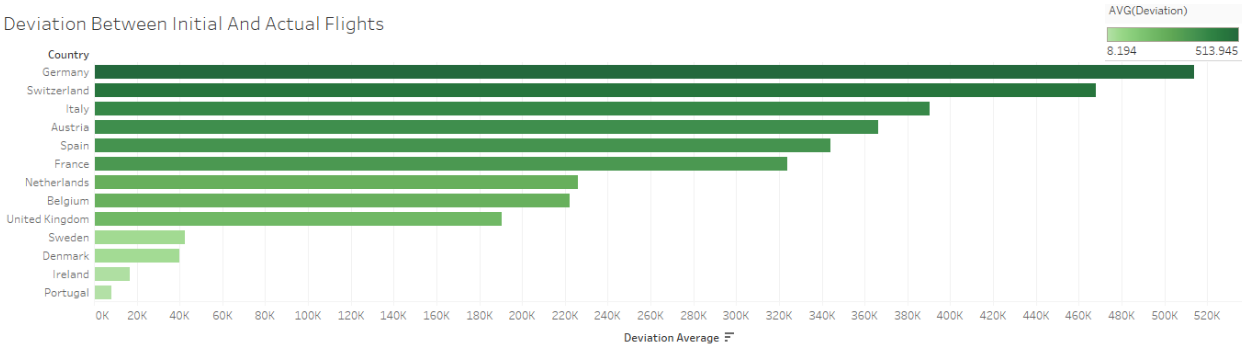

Figure 4.14: EWG Cost variation between planned and real flights per country on 9th May 2019

Graphic 4.14 proves the tool benefits for this airline. In conclusion, all the airspaces where Eurowings operate show a beneficial impact for the airline. It means that the real flights routes are more direct giving importance to countries such as Germany, Switzerland and Italy.

In countries such as Sweden, Denmark, Ireland and Portugal do not have as much influence in costs savings. The reasons are that these airspaces do not have as much air traffic inflow and there is not much frequency of routes daily operated by this airline.

Relative error compared to NEST application

In this part, 28 flights with different origins and destinations are analysed, limited to the airspace scenario of this first version and countries in which the airline operate. The error is shown compared to the NEST application and the analysis of the most affected routes.

To compare the error between NEST application and script En-Route Charges Calculation, the equations described in 4.2. are applied.

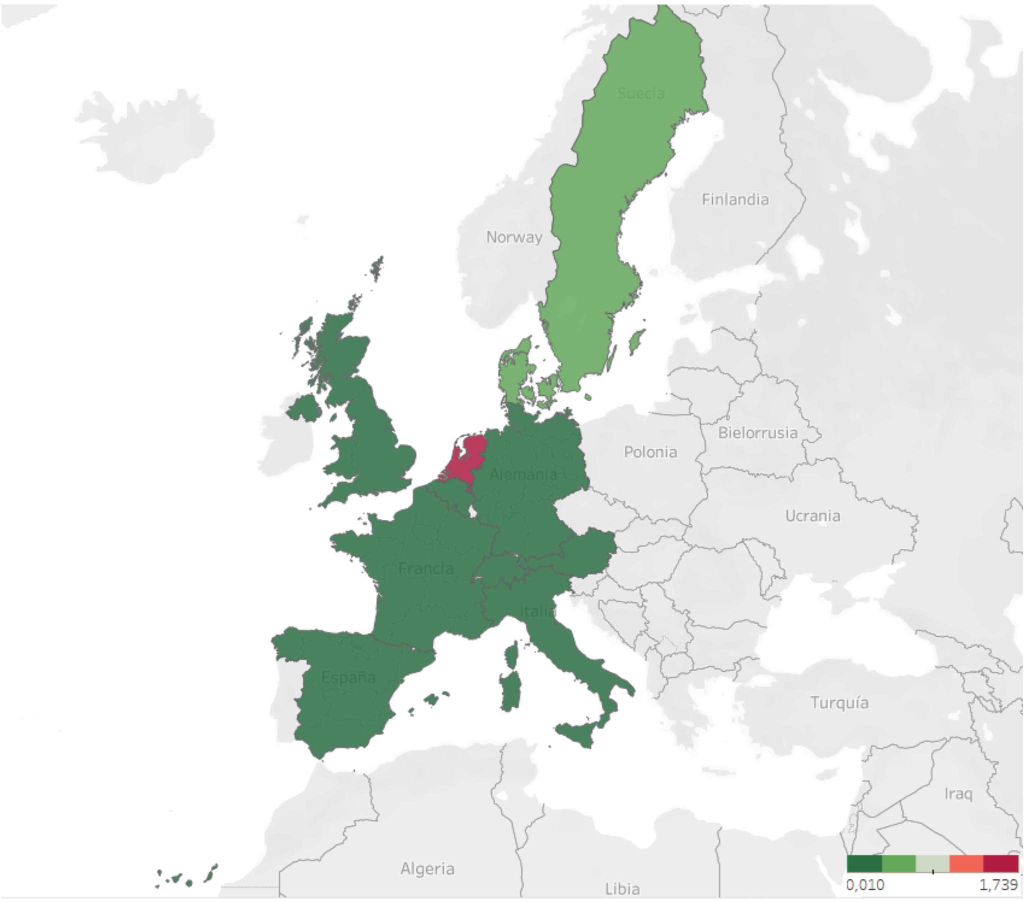

Figure 4.15: EWG Representation of Relative Error with NEST

Map 4.15 shows the countries with a greater error than the other countries where this airline operates. In general conditions, the relative error is small enough to validate the results except for the country The Netherlands.

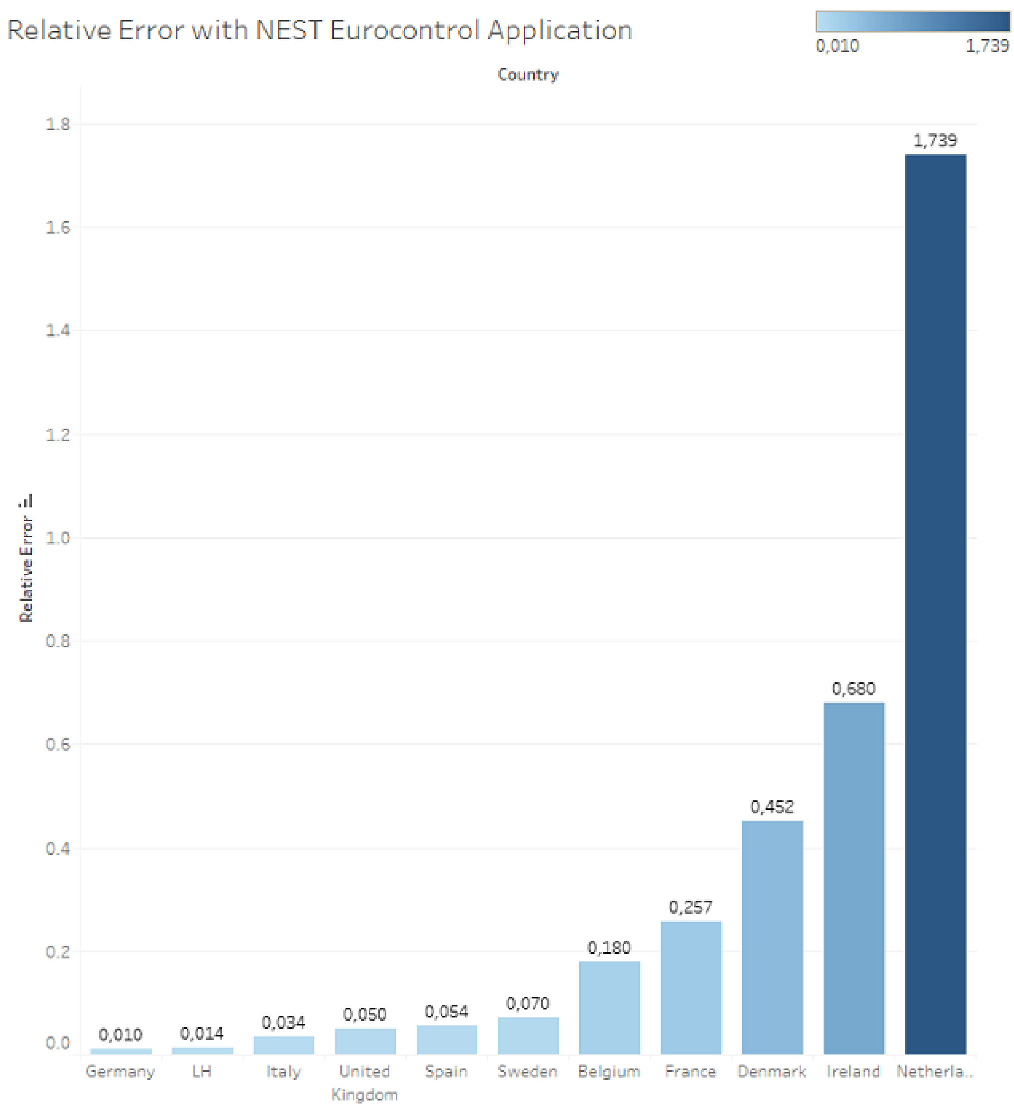

Figure 4.16: Relative Error per country of EWG airline

With graphic 4.16, it can be analysed the error per country in detail. As previously mentioned, The Netherlands is the country with the less reliable results. The reason is because of the flight samples we have chosen, there are not many of them that cross this airspace.

Also, the error may be due to the coordinates. The coordinates have been studied from the AIP of each country and the boundaries, although there may be a slight difference with those defined by NEST, as they are not available.

As the relative error is small for certain countries and with the same magnitude comparing it with the other airlines, these relative errors can be neglected.

Analysis of Eurowings routes

Finally, in this section, the different routes are analysed in order to study which one have a more cost variation between the planned and real route. Then, the reasons are justified and some conclusions can be reached.

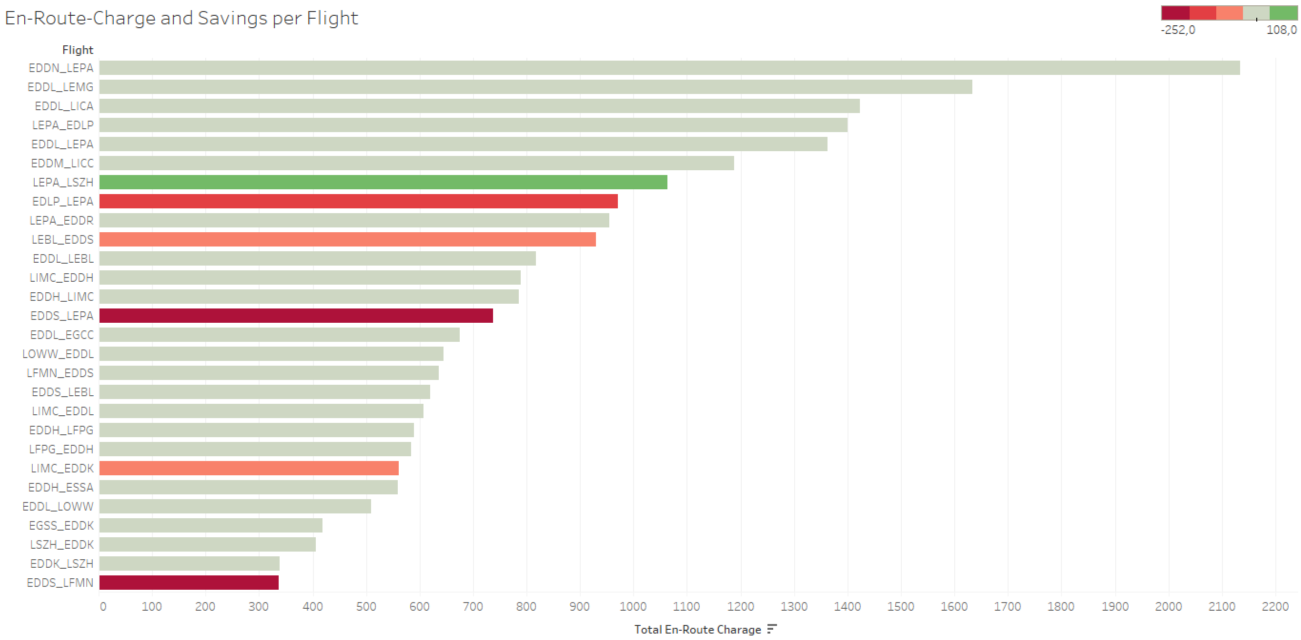

Figure 4.17: En-Route Charges per EWG Flights

Graphic 4.17 shows the total en-route charge of EWG flights. In this case, 28 flights are analysed and plotted in this graph. As the other airlines, almost all the flights do not have an important cost deviation but in this case, there is a greater number of flights affected and in a negative way for the project, therefore, the initial trajectories are cheaper.

This is not a problem since the main and most expensive routes do not have much deviation. What can be observed again is the fact that almost all flights bound for or crossing Germany are affected, again.

Finally, the route with the worst cost difference EDDS-LFMN is also the most economical route.

4.4.4. EasyJet

EasyJet is a low-cost British airline based at London Luton Airport that operates 830 routes throughout Europe and airports in North Africa. Its main airport is London Gatwick.

EasyJet Global Analysis

Firstly, a global assessment of the airline is shown. Furthermore, it will be able to analyse which countries are most affected and why.

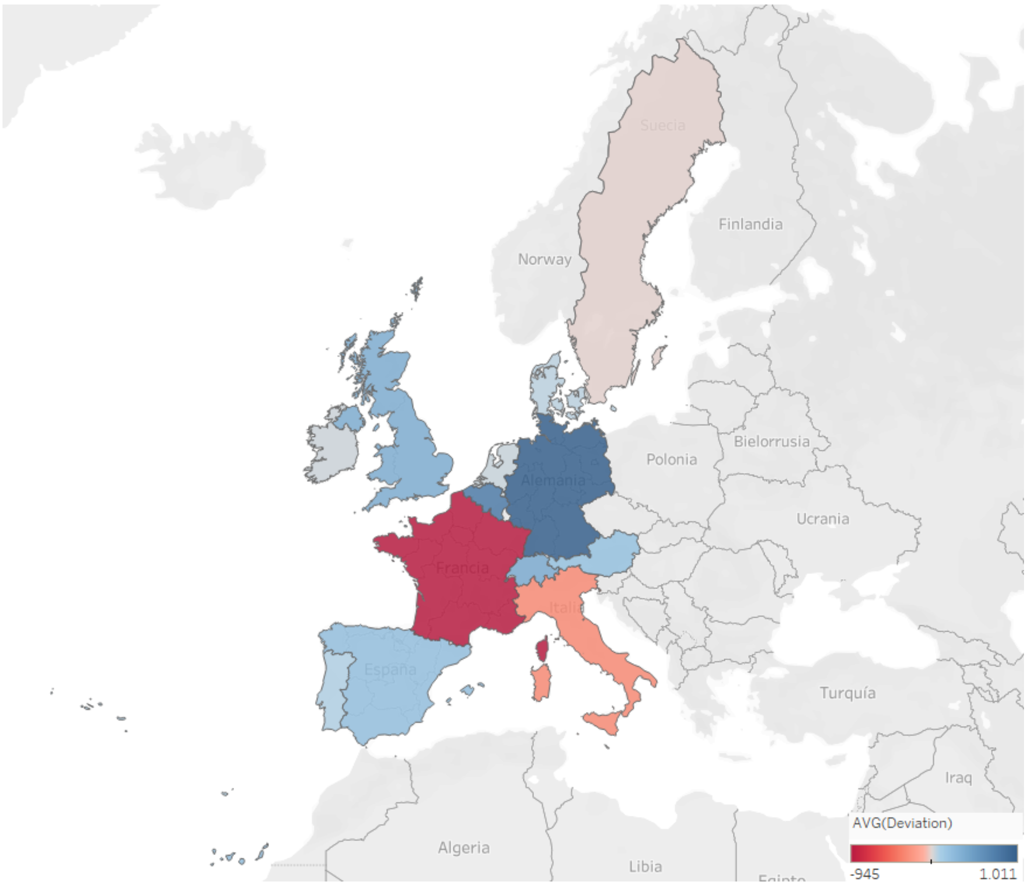

Figure 4.18: EZY Representation of cost variation between planned and real flights on 9th May 2019

In this case, the France restriction impact can be recognized by this map. About 65% of its flights pass through French airspace. Moreover, their daily routes are also centered, they are more numerous in Italy, although its maximum negative value is not as important as can be seen on the map (-945€).

The values of the positive range are not very high either, but almost the whole scenario of the first tool version shows a beneficial difference. The country of Germany can be highlighted again in terms of profits due to considering the real flight.

In figure 4.19, it is assessed these cost differences in numbers.

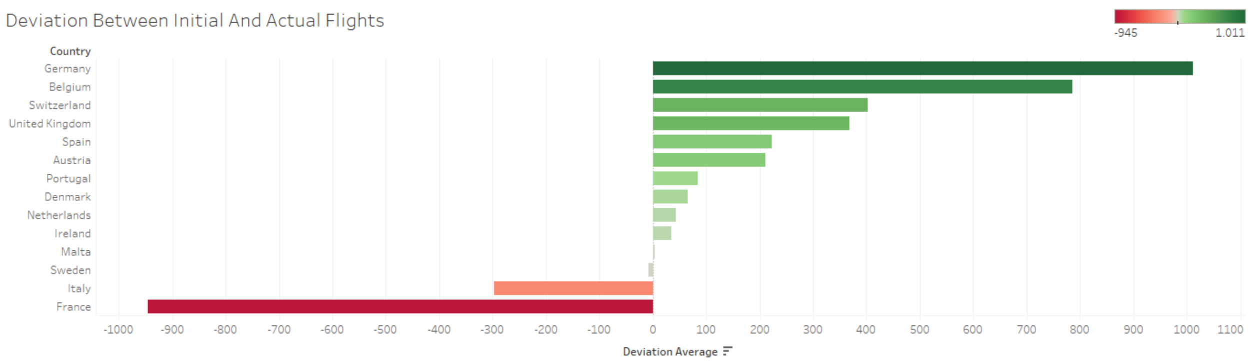

Figure 4.19: EZY Cost variation between planned and real flights per country on 9th May 2019

Graph 4.19 shows us again the differences between countries, where almost all have positive results. In this case, it can be recognized that the countries with the most difference, whether positive or negative, are the most transited by this airline.

In conclusion, EasyJet Airline, as Eurowings, show a beneficial impact in terms of reduction of Navigation Taxes. This airline has more costs due to the consideration of the initial flights, according to the scenario studied.

Relative error compared to NEST application

In this section, 29 flights with different origins and destinations are analysed, limited to the airspace scenario of this first version and countries in which the airline operate. The error is shown compared to the NEST application and the analysis of the most affected routes.

To compare the error between NEST application and script En-Route Charges Calculation, the equations defined in section 4.2. are applied.

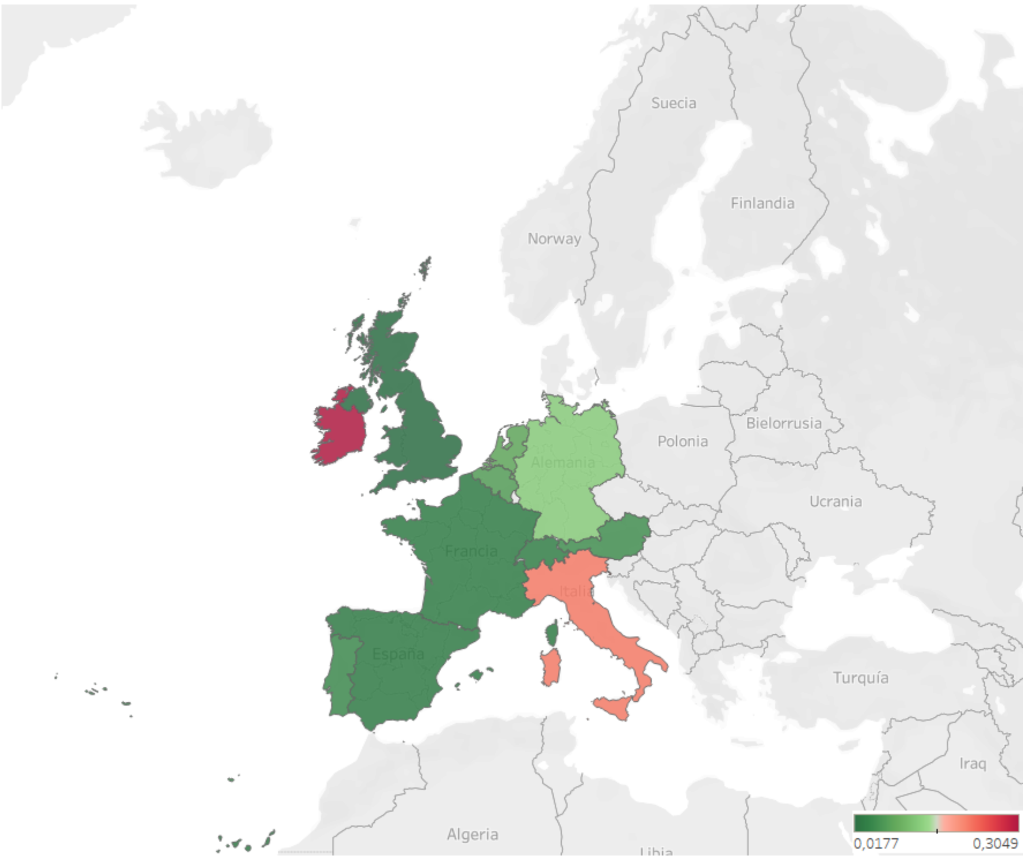

Figure 4.20: EZY Representation of Relative Error with NEST

Map 4.20 shows the countries with a greater error than the other countries where this airline operates. In general conditions, the relative error is small enough to validate the results except for the Ireland and Italy airspaces, although their mistakes are very slight.

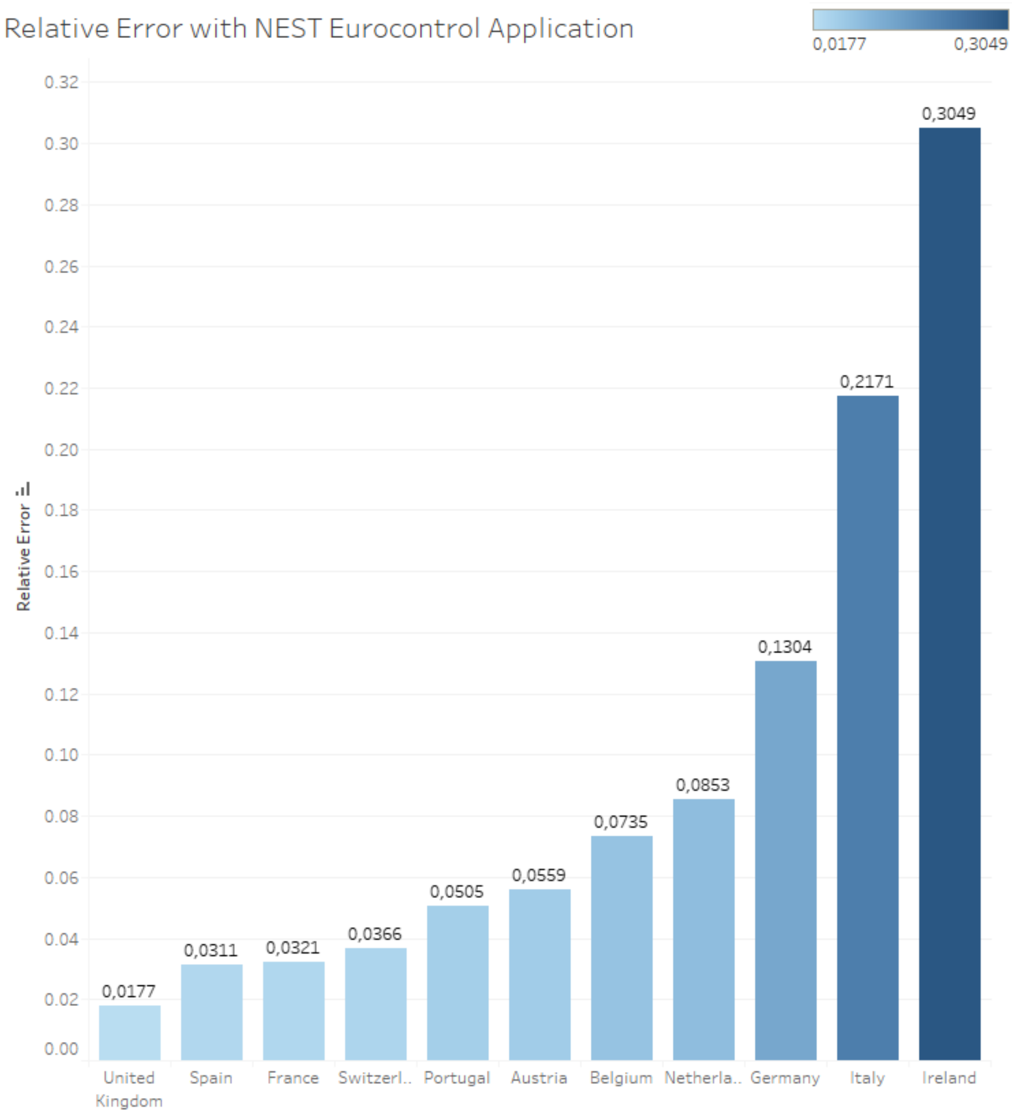

Figure 4.21: Relative Error per country of EZY airline

With graphic 4.21, it can be analysed the error per country in detail. As previously mentioned, Ireland and Italy are the country with the worst results. The error may be due to the coordinates.

The coordinates have been studied from the AIP of each country and the boundaries, although there may be a slight difference with those defined by NEST, as they are not available. Another reason is the quantity of flights applied in the simulations that crosses these countries.

Again, the principal countries with smaller relative error coincide with the previous cases of study.

Analysis of EasyJet routes

Finally, in this section, the different routes are analysed in order to study which one have a more cost variation between the planned and real route. Then, the reasons are justified and some conclusions can be reached.

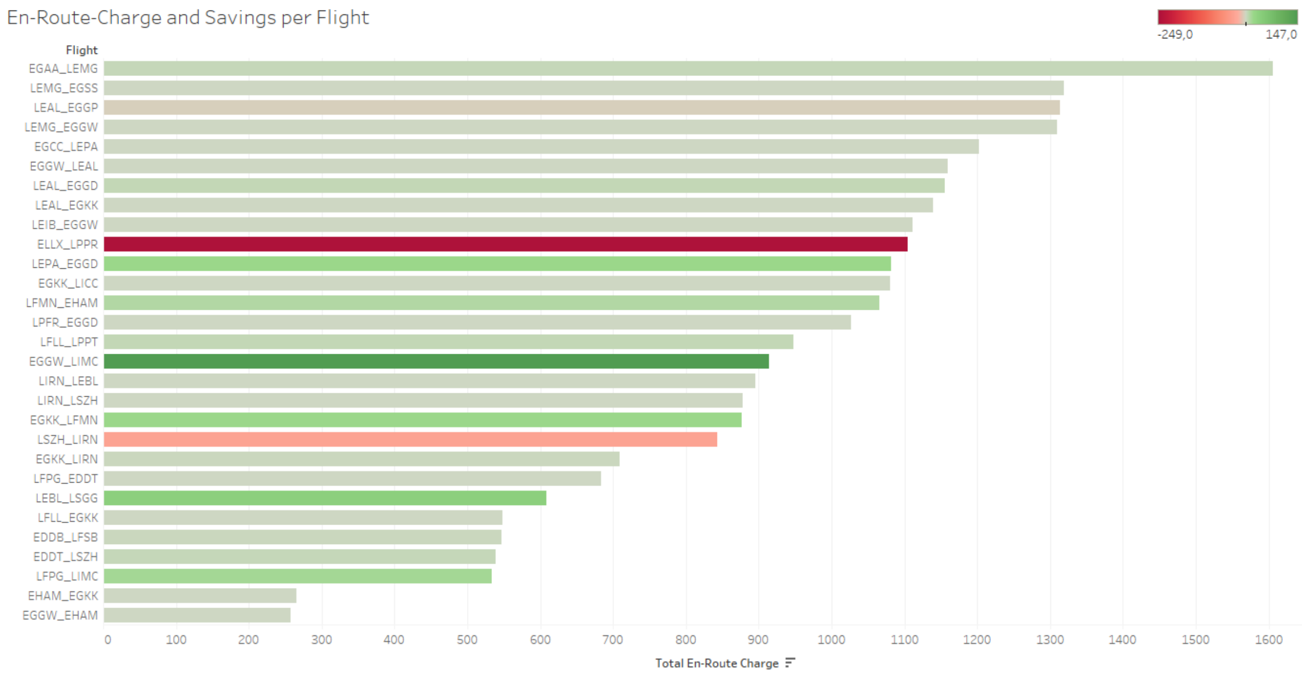

Figure 4.22: En-Route Charges per EZY Flights

As graphic 4.22 shows, and unlike other airlines, almost all flights show a small difference in cost, positive or negative. This shows that the actual flights of this company differ from the flight plans.

The tendency of these flights is to have a positive cost difference, as we can see a graph where the green prevails.

Moreover, the flights have higher en-route charges values compared to other airlines, this may be because their flights are not as efficient or direct as the other airlines. The simulations are made on a day when France's airspace was restricted, its main flights usually fly through this country and therefore, this can be an important factor which influences the results.

4.4.5. Vueling Airlines

Vueling Airlines is a Spanish airline, based in Barcelona. It is the largest airline within Spanish territory in number of destinations and by fleet size, and the second by number of passengers transported within Spanish territory.

Vueling's main base since its creation and also its hub since May 2010 (for further information see reference[44]) is in Barcelona, while it has numerous bases of operations in the rest of Spain.

Vueling Global Analysis

Firstly, a first global analysis of the cost variation is carried out to show if this airline would benefit from the application. Furthermore, it will be able to analyse which countries are most affected and why.

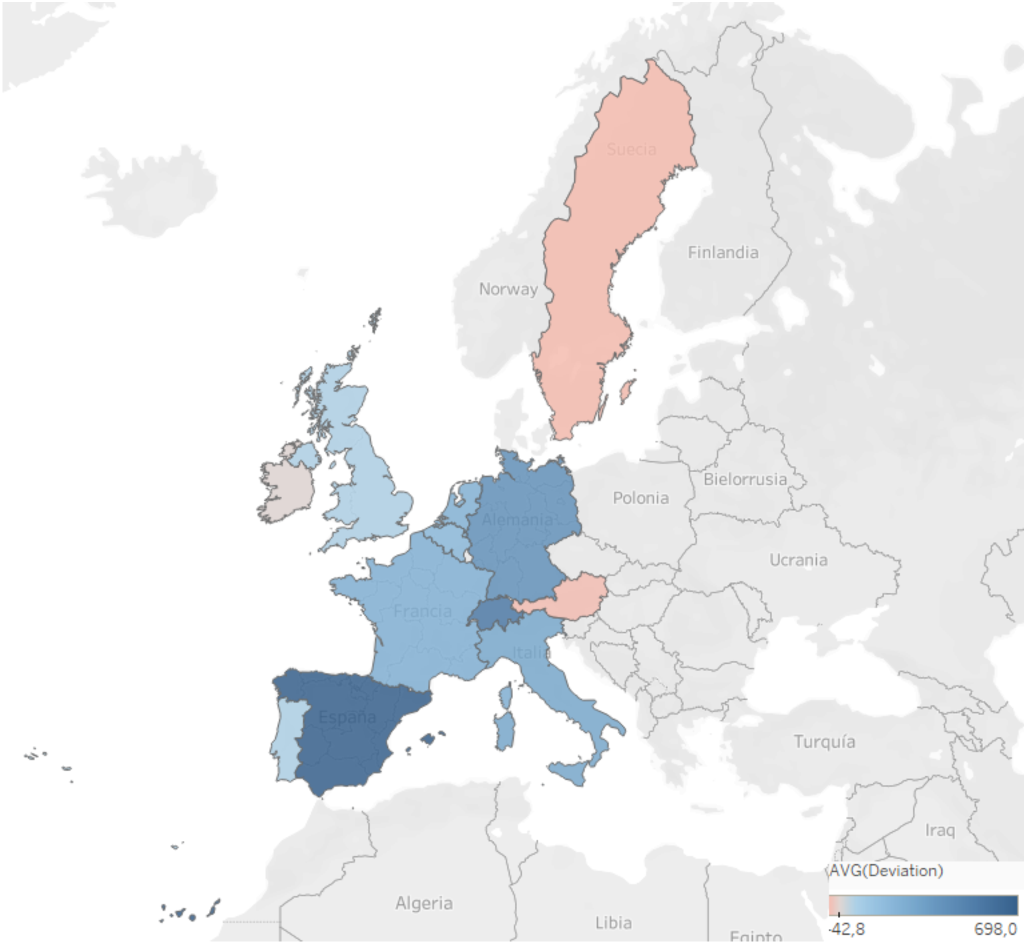

Figure 4.23: VLG Representation of cost variation between planned and real flights on 9th May 2019

As shown on map 4.23, the main countries of operation are those that show a greater difference, in this case positive and therefore proves that the actual flight plans are more efficient and direct than those planned, specially focused on Spain.

In general terms, Vueling has a global positive cost variation in Europe. The countries with negative impact are for example Sweden or Switzerland, although as we can observe, the maximum negative range is very low, hardly negligible.

Figure 4.24 shows these cost differences in numbers.

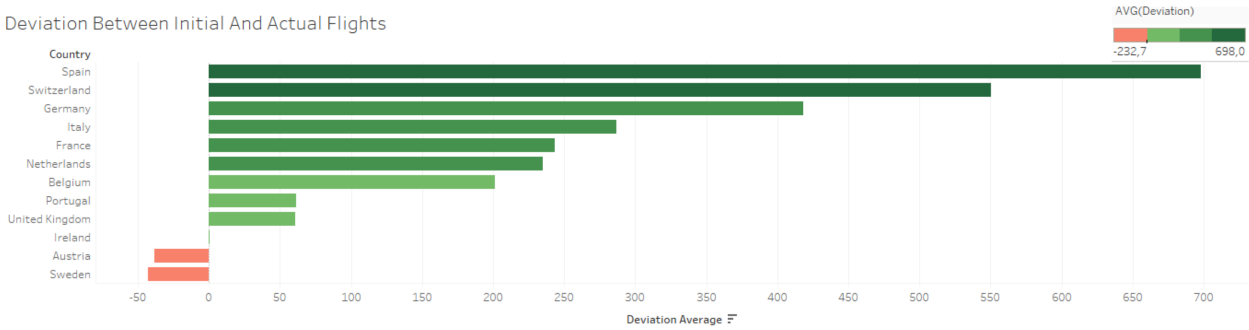

Figure 4.24: VLG Cost variation between planned and real flights per country on 9th May 2019

The graphical representation of the cost difference in numbers is shown in figure 4.24. It proves that Vueling would have lower navigation costs if it was considered the actual flights to compute the en-route charges.

The main countries are Spain, by the number of Vueling flights in this region, Switzerland, as it is the area with the highest unit rate, and finally Germany, that matches with all the other airlines studied.

Austria and Sweden have a negative impact, but their air traffic is not as much heavy as other countries and Vueling's operations in these countries are limited.

Relative error compared to NEST application

In this section, 20 flights with different origins and destinations are analysed, limited to the airspace scenario of this first version and countries in which the airline operate. The error is shown compared to the NEST application and the analysis of the most affected routes.

To compare the error between NEST application and script En-Route Charges Calculation, the formulas described in 4.2. are applied.

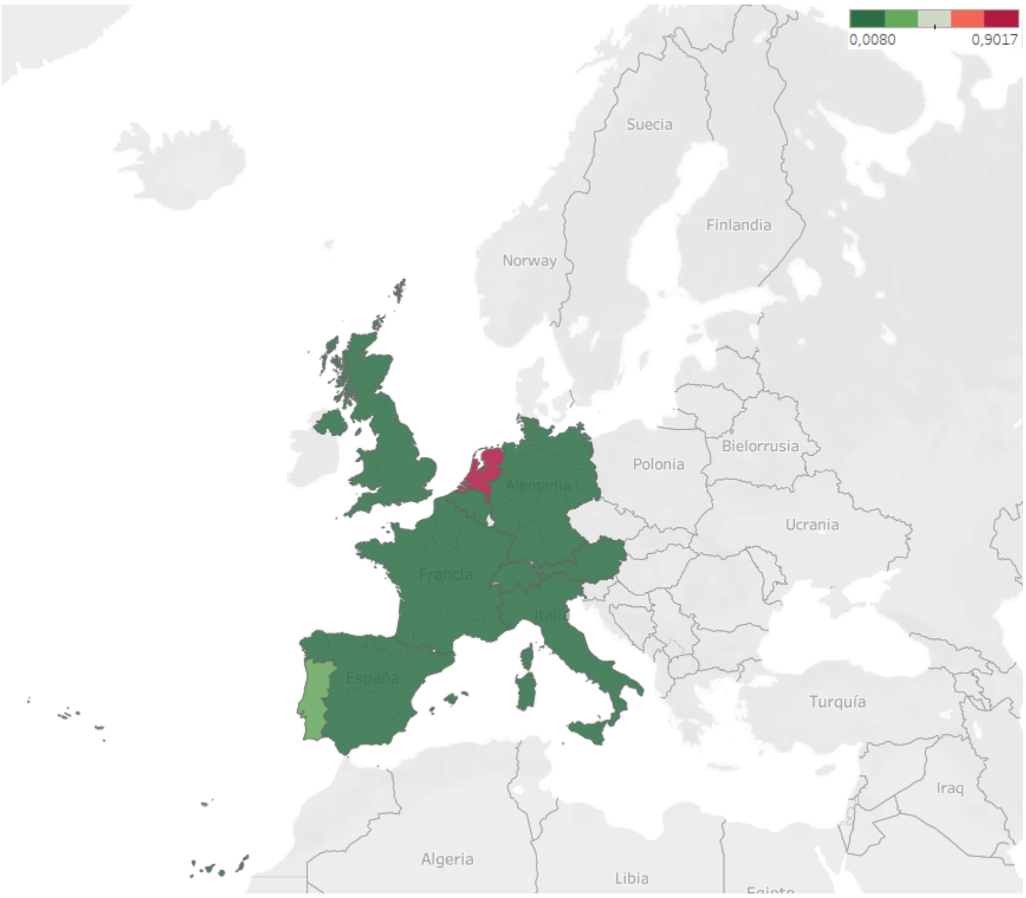

Figure 4.25: VLG Representation of Relative Error with NEST

Map 4.25 shows the countries with a greater error than the other countries where this airline operates. In general conditions, the relative error is small enough to validate the results except for the The Netherlands, although their mistakes are very slight.

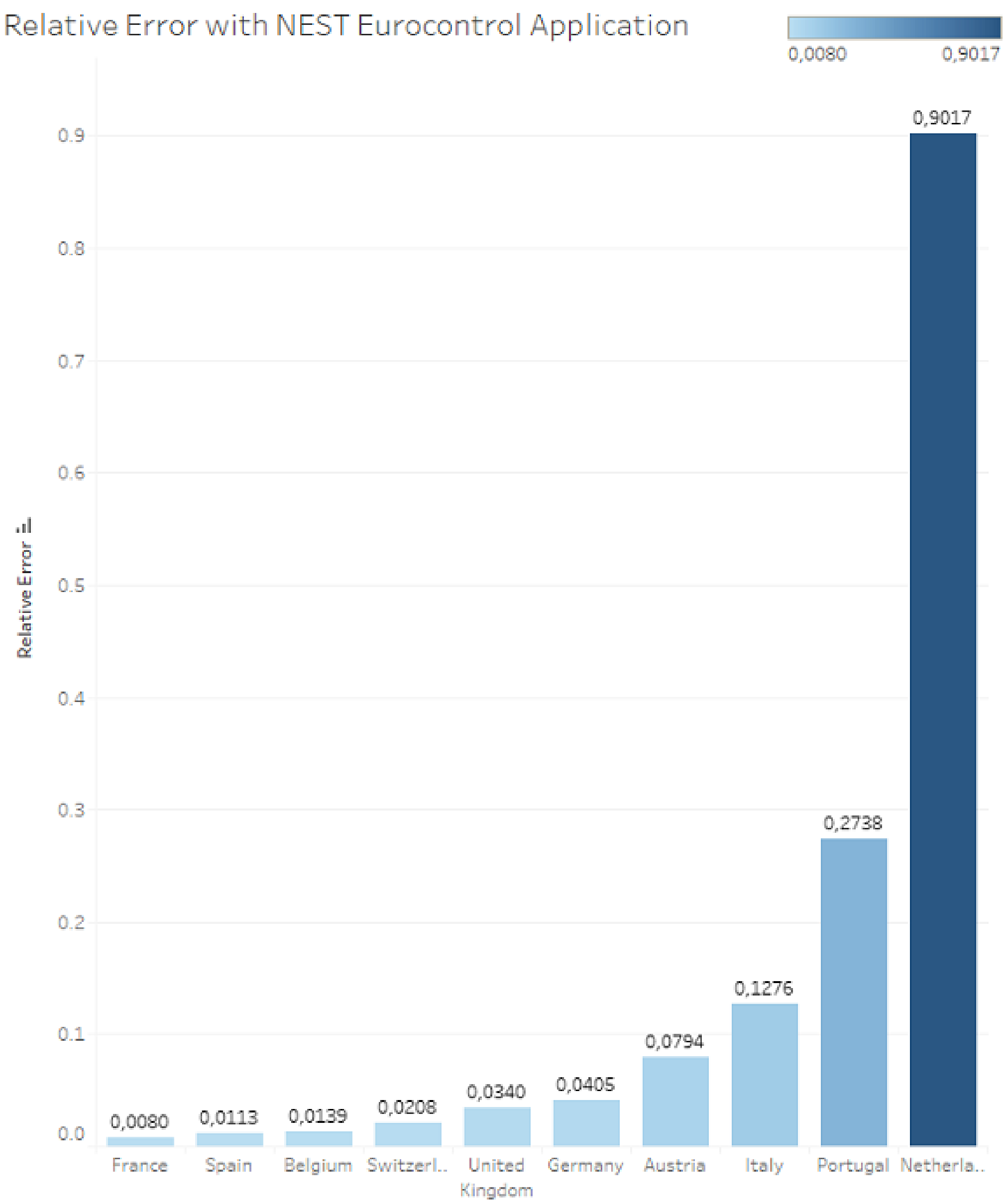

Figure 4.26: Relative Error per country of VLG airline

With graphic 4.26, it can be analysed the error per country in detail. As previously mentioned, The Netherlands the country with the worst results. The reason is the few flights that fly over this country and that therefore not so many samples are found. With a larger number of samples, the error can be computed more accurately.

Finally, the principal countries with a very low error and that allows to validate the results, coincides with the results of the airlines analysed in this project. Therefore, this justifies that the error is given by some countries where their airspace must be profiled, and which have less air traffic according to the airline.

Analysis of Vueling routes

Finally, in this section, the different routes are analysed in order to study which one have a more cost variation between the planned and real route. Then, the reasons are justified and some conclusions can be reached.

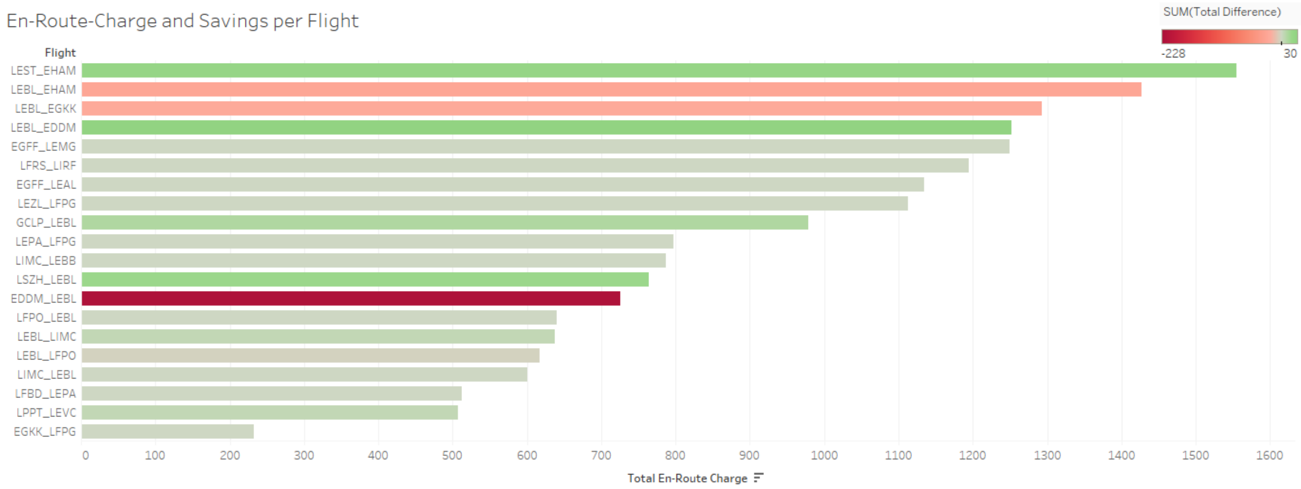

Figure 4.27: En-Route Charges per VLG Flights

Finally, for Vueling, the graphic 4.27 shows that the results are not so beneficial. Although there are a greater number of flights with positive differences, the range is much larger for the negative axis than for the positive axis. This is because French airspace was restricted and Spanish airspace was directly affected. Vueling has different routes mainly at national level but also at international level. France is the neighbouring country and many of its flights are conditioned and changed by it, as they fly through French airspace.

As a final remark, the first and fourth flight with high en-route charges have positive difference. The impact of a profit in a more expensive flight is more important than on shorter flights.

CONCLUSIONS AND RESULTS

To conclude this project, this section describes the results of the project, performing a global analysis.

As first important point to emphasize, the main objectives of this project have been achieved.

First, the difference in costs between flight plans and real routes has been demonstrated, in the global and airline study (sections 4.3., 4.4.). This cost variation shows that the cost of en-route charges for actual (real) flights implies a reduction of DOCs Navigation Taxes.

Therefore, implementing this technology would benefit airline DOCs management, although the implementation of this case study should be defined with Eurocontrol, which manage the billing and recovery of Air Navigation charges.

Secondly, the relative error between the results obtained with this first tool version and NEST en-route charges are computed by country.

In the Airlines Global Study, it has been proved that the error is not very significant, but some countries are affected as Sweden, Netherlands, Ireland and Italy. The reasons would be caused by:

- Airspace coordinates: the airspaces defined in this project are computed based on countries AIP because it has not been possible to achieve the same airspace sectors of Eurocontrol application NEST. This difference could cause the slight errors in these countries.
- Flight coordinates: trajectories are extracted from NEST but the data could have a minimum percentage of error. Apart from that, coordinate points are defined by navigation points which means that it is not fully tracked in detail.
- Number of flights simulated: some airspaces are more congested than others and the simulations have been based on a specific day and on selected airlines and flights. Therefore, the number of samples crossing some countries are few and therefore does not allow us to make a proper error calculation in these airspaces.

All these reasons are factors of improvement for future versions of the tool, as mentioned in the further implementations section.

Regarding the Global Airline Analysis 4.3., it can be seen how in general all countries have beneficial results mainly in Germany, Belgium, Ireland and Switzerland. The reason why these countries get more profits is because air traffic controllers, in these countries, order more direct routes at the time of flight and consequently the real routes are more efficient than initial routes.

The air traffic controllers order more direct routes in these countries due to the large number of flights, the congestion of these airspaces. Moreover, the flights connecting these countries have origin from and destination to the most important airports of Europe.

In contrast, countries such as United Kingdom, Spain, France and Italy show negative cost different which means that initial flights are more economic. The day chosen to perform the simulations was 9th of May. On this day, France airspace was restricted, and its operations limited. The surrounding airspaces, as proved in airlines analysis (section 4.4.), are directly affected by this problematic, as they had to support air traffic that could not fly through France. Their initial routes and all their scheduling changed.

Finally, it has also been possible to analyse which airlines have more efficient flights and which routes show more cost deviation between initial and actual routes. Eurowings airline is the one that obtains worse results, its planned routes suppose less cost than the real ones. By contrast, EasyJet is the airline which obtains better results, this first tool version is beneficial for EasyJet.

In addition, this project has a future development plan, described in future implementations section, to improve its functionality. In this way, a very powerful final tool could be realised for the airlines and for the sector in general.

In conclusion, air traffic management must improve and create systems in order to sustain the growing air traffic demand. Therefore, it will be possible to build a much more efficient and economical airspace in which this transport system will be more beneficial and advantageous for users.

FURTHER IMPLEMENTATIONS

Finally, the different future implementations and improvements of this project are proposed and established, some of which have already been discussed in previous sections.

First, the airspace scenario of this first tool version includes 17 of the 41 EUROCONTROL State Members. One important proposal is to create a complete scenario where all airspaces are integrated and available. Therefore, the tool will have a full coverage.

In consequence, the future tool version could support all the flights without excluding any aircraft type, any country where it crosses or any airline. Also, the Airplane Solutions prototype will be installed in coming months and therefore, the flight data from aircraft will be processed by the script, as the NEST data. Furthermore, the trajectories will be more accurate because the latitude and longitude would be available every minute.

Moreover, the Airplane Solutions technology compute the DOCs taxes and fees using Blockchain technology. Nowadays, this project is computed using Python script. In a second phase of this project, the aim is to calculate the en-route charges using this technology.

Additionally, Airplane Solutions create a web user to plot the results and costs of all other fees. In this project, the graphical representation is performed using Carto Platform and Tableau. As a future implementation, these plots and graphs will be shown in Airplane Solutions web user platform. Also, each flight will be identified per airline, day and hour and origin and destination countries.

Finally, there will be a correction and reduction of calculation errors studied in previous sections regarding for example the numerical error that the coordinates may introduce to the tool.

BIBLIOGRAPHY

[1] (2018). Airbus Global Market Forecast 2018-2037 (book). AIRBUS. Web site: https://www.airbus.com/aircraft/market/global-market-forecast.html 3

[2] (2018). EUROPEAN AVIATION IN 2040. CHALLENGES OF GROWTH. EUROCONTROL. Web site: https://www.eurocontrol.int/sites/default/files/content/documents/official-documents/reports/challenges-of-growth-2018.pdf 3

[3] (2nd June 2016). Brian Pearce - IATA Economics. ECONOMIC PERFORMANCE OF THE AIRLINE INDUSTRY. IATA. Web site: https://www.iata.org/whatwedo/Documents/economics/IATA-Economic-Performance-of-the-Industry-mid-year-2016-report.pdf

[4] (2019). Skywise. AIRBUS Web site: https://www.airbus.com/aircraft/support-services/skywise.html 7

[5] ABOUT DISCOVER SESAR. SESAR JOINT UNDERTAKING. Web site: https://www.sesarju.eu/ 7

[6] INDRA AIR INFORMATION. INDRA. Web site: https://www.indracompany.com/es/indra-air-information 7

[7] Alessia Violin, Martine Labbé, Lorenzo Castelli, 2010/6/1. ICRAT 2010: 4th International Conference on Research in Air Transportation, 433-440. En route charges for ANSP revenue maximization. Editors: Vu Duong, David J.Lovell. Web site: http://www.icrat.org/icrat/4th-international-conference/icrat_2010_proceedings.pdf 7

[8] (February 2019). Central Route Charge Office Customer Guide to Charges. EUROCONTROL. Web site: https://eurocontrol.int/sites/default/files/2019-05/customer-guide-charges-20-feb-2019_0.pdf 8

[9] (January 2019) CENTRAL ROUTE CHARGES OFFICE. *Information Circular n^{o}2019/02, Distribution*. European Organisation for the Safety of Air Navigation (EUROCONTROL) 8

[10] (24 MAY 2019). European Civil Aviation Conference. WIKIPEDIA. Web site: https://en.wikipedia.org/wiki/European_Civil_Aviation_Conference 10

[11] (10-11 APR 2014). The SES Performance Scheme, ICAO Regional Performance Framework Workshop. SINGLE EUROPEAN SKY. 10

[12] (2019). Network Manager. EUROCONTROL - Supporting European Aviation. Web site: https://www.eurocontrol.int/network-manager 10

[13] Central Route Charge Office. *Document 716028 Tables in Excel Format, A-8 En-Route exemptions (as from 01/10/2017)*. European Organisation for the Safety of Air Navigation (EUROCONTROL). 10

[14] (June 2018). Central Route Charge Office. *Customer Guide to Charges*. European Organisation for the Safety of Air Navigation (EUROCONTROL). 8, 11, 12, 13

[15] Great Circle Route, Time and Navigation. SMITHSONIAN INSTITUTION.

Web site: https://timeandnavigation.si.edu/multimedia-asset/great-circle-route 11

[16] Great circle distance. Wikipedia Foundation 2018. Web site: https://en.wikipedia.org/wiki/Great-circle_distance 11

[17] Maximum Take-Off Mass - (MTOM). Skybrary Wiki Platform. Web site: https://www.skybrary.aero/index.php/Maximum_Take-Off_Mass_(MTOM) 12

[18] (2019) Monthly Adjusted Unit Rates, Route Charges. EUROCONTROL - Supporting European Aviation. Web site: https://www.eurocontrol.int/services/monthly-adjusted-unit-rates 13, 21

[19] (December 2018). Central Route Charge Office. *EUROCONTROL Route Charges System, Information Circular Nº 2019/01 to Users*. European Organisation for the Safety of Air Navigation (EUROCONTROL).

[20] (November 2015). EUROCONTROL Bussines Case Team. *Standard Inputs for EUROCONTROL Cost-Benefit Analyses.* Edition Number: 7.0. European Organisation for the Safety of Air Navigation (EUROCONTROL).

[21] En-Route Airspace. Eurocontrol ATM Lexicon. Web site: https://ext.eurocontrol.int/lexicon/index.php/En-route_airspace 15

[22] El espacio aéreo, Gaceta Aeronautica. Web site: https://www.gacetaeronautica.com/gaceta/wp-101/?p=5126 15

[23] (2019). FIR/UIR charts. EUROCONTROL - Supporting European Aviation. Web site: https://www.eurocontrol.int/articles/firuir-charts 16

[24] (WEF 08-NOV-18). AIP ESPAÑA, ENR 2.1-1. ENAIRE. Web site: hhttps://ais.enaire.es/AIP/AIPS/AMDT_311_2019_AIRAC_01_2019/AIP.html 17

[25] (2018). AIP PORTUGAL, ENR 2.1. FIR AND TMA. NAV Portugal, E.P.E. Web site: https://www.nav.pt/CD/NAVProtectedData/eAIP/html/index.html 17

[26] (WEF 03-APR-2014). AIP MALTA, ENR 2.1-1. TM. Web site: https://ec.europa.eu/transport/sites/transport/files/fraLM_Circ_2016_A_02_en.pdf 17

[27] (WEF 24-MAY-2018). AIP ITALIA, ENR 2.1,1-1. ENAV. Web site: https://www.enav.it/enavWebPortalStatic/AIP/AIP/enr/enr2/ENR2.htmf 17

[28] (AIRAC AMDT 04/18). AIP FRANCE, ENR 2.1-2. DGAC. Web site: https://www.sia.aviation-civile.gouv.fr/dvd/eAIP_20_JUN_2019/FRANCE/AIRAC-2019-06-20/html/index-fr-FR.html 17

[29] (02 MAR 2017). AIP SWITZERLAND, ENR 2.1. SKYGUIDE. Web site: http://aip.engadin-airport.ch/eAIP/2017-06-22/html/index.html 17

[30] (13 SEP 2018). AIP AUSTRIA, ENR 2.1. AUSTRO CONTROL.Web site: https://eaip.austrocontrol.at/lo/181207/PART_2/LO_ENR_2_1_en.pdf 17

[31] (17 AUG 2017). AIP GERMANY, ENR 2.1. DFS. Web site: http://www.ead.eurocontrol.int/eadbasic/pamslight-DF8F927A9CA611510F07E527857C0549/OODUMTUY4LXFU/EN/AIP/ENR/ED_ENR_2_1_en_2019-06-20.pdf 17

[32] AIP for BELGIUM, ENR 2.1. Skeyes. Web site: https://ops.skeyes.be/html/belgocontrol_static/eaip/eAIP_Main/html/eAIP/EB-ENR-2.1-en-GB.html

17

[33] (23 MAY 2019). UNITED KINGDOM AIP, ENR 2.1-1. NATS. Web site: http://www.ead.eurocontrol.int/eadbasic/pamslight-F6D62265B3273661B2122321AFBA5C90/7FE5QZZF3FXUS/EN/AIP/ENR/EG_ENR_2_1_en_2019-05-23.pdf 17

[34] (08 NOV 2019). AIP IRELAND, ENR 2.1. IAA. Web site: http://iaip.iaa.ie/iaip/IAIP_Frame_CD.htm 17

[35] (20 JUN 2019). AIP NETHERLANDS, ENR 2.1. LVNL. Web site: https://en.lvnl.nl/eaip/2019-06-06-AIRAC/html/index-en-GB.html 17

[36] (28 MAR 2019). AIP SWEDEN, ENR 2.1. LFV. Web site: https://aro.lfv.se/Editorial/View/6094/ES_ENR_2_1_en 17

[37] (20 JUN 2019). AIP NORWAY, ENR 2.1-1. AVINOR. Web site: https://ais.avinor.no/no/AIP/View/58/aip/EN_ENR_2_1_en.pdf 17

[38] (2019). GADM maps and data. EUROCONTROL - Supporting European Aviation. Web site: https://gadm.org/index.html 18

[39] TOOL AIRSPACE SCENARIO IN CARTODB. Web site: https://annaaru.carto.com/builder/ab6706fd-72db-4631-a134-feed1b9de7ca/layers 19, 22

[40] Difference between quantitative and qualitative research. SURVEY MONKEY. Web site: https://es.surveymonkey.com/mp/quantitative-vs-qualitative-research/ 25

[41] (December 2003). ERROR PROCESSING AND PROPAGATION. Web site: https://www.lawebdefisica.com/apuntsfis/errores/ 27

[42] TABLEAU. Web site: https://www.tableau.com/es-es/products/desktop 27

[43] (January 14, 2019). Lufthansa Becomes Europe's Largest Airline. SIMPLE FLYING. Web site: https://simpleflying.com/lufthansa-europes-largest-airline/ 33

[44] (Oct 7, 2013). Vueling Refines Barcelona Hub With Shorter Connections, AVIATION WEEK NETWORK. Web site: https://aviationweek.com/awin/vueling-refines-barcelona-hub-shorter-connections 47

[45] EN-ROUTE CHARGES CALCULATION REPOSITORY https://gitlab.com/annaa_ru/en-route-charges-calculation 63

APPENDICES

APPENDIX A. SCRIPT: EN-ROUTE CHARGE CALCULATION

Script available in GitLab repository (for further information see reference [45]):

Main.py

```
from Scenario import Flight, Airspace
from Intersection import Intersection
from EnRouteCharge import EnRoute
from Ddbb import Postgresql
from SetLabels import Labels
import pandas as pd
import cartoframes
from carto.auth import APIKeyAuthClient
import webbrowser
import csv

#Insert the airplane type folder
model = "A320"
#Insert the airline folder
airline = "EWG"
#Insert the flight:
flight = "LEBL_EDDS"

#Connection to CARTO DB API
USERNAME = "annaaru"
USR_BASE_URL = "https://{user}.carto.com/".format(user=USERNAME)
auth_client = APIKeyAuthClient(api_key="
    ... f3ed48b92fc3a93e1cb452da2454bf9d8e117b35", base_url=
    ... USR_BASE_URL)

BASE_URL = "https://{organization}.carto.com/user/{user}/". \
    format(organization="annaaru",
          user="annaaru")

#Read the Excel Sheets, to diferenciate between Initial and
    ...  Actual Flight
path = '../Coordinates/{0}/{1}/{2}.xlsx'.format(model, airline,
    ... flight)
excel = pd.ExcelFile(path)
length = len(excel.sheet_names)

#Create the classes
Airspace = Airspace()
Intersection = Intersection()
EnRoute = EnRoute()
Labels = Labels()
```

```python
#Iteration of the different flights (in this case, Initial
    ... and Actual Trajectory of the same flight)
i = 0
while i <= (len(excel.sheet_names)-1):
    #Create a LineString from flight coordinates
    sheet = excel.sheet_names[i]
    F = Flight()
    F.get_flight_coordinates(path, sheet)

    #Create a Route to define the countries involved
    F.Route = Airspace.get_countries(F.listFlight)

    #Get the Countries Boundaries Polygon from previous
        ... Route
    Airspace.get_polys(F.Route)

    #Computation of intersection points between countries
        ... and flight
    Intersection.get_intersection(F.FlightCoord, Airspace.polys, F.
        ... Route, F.listFlight)
    Intersection.get_en_route_points(F.listFlight)

    #Connection to APS data base to get the Monthly Adjusted
        ...  Unit Rates from Eurocontrol website
    Sql = Postgresql()
    Unit_rates = Sql.get_unit_rates(F.Route)
    Mtom = Sql.get_mtom()
    Mtom = round(float(Mtom[0][0]), 1)

    #Get the specific Unit Rates from the countries involved
    Factors = EnRoute.get_unit(Unit_rates, F.Route)

    #Distance Calculation between intersection points
    d = EnRoute.get_distance(Intersection.enRoutePoints, Factors)

    #Final Calculation of EN-ROUTE-CHARGES taking account
        ... the formula described by Eurocontrol
    Charges = EnRoute.get_enroute_charge(EnRoute.EnRouteFactors, Mtom)

    #Extraction of the flight and charges information for
        ... export to Carto DB
    Data = Labels.set_labels(Intersection.enRoutePoints, Charges, d)
    CsvData = Labels.create_csv(Intersection.enRoutePoints, sheet,
        ... flight)

    i = i + 1
```

```
#Connection to Carto DB to export all the needed
    ... information
ds = pd.read_csv('../SRC/Initial.csv')
df = pd.read_csv('../SRC/Actual.csv')
cc = cartoframes.CartoContext(base_url=USR_BASE_URL,
                              api_key="
                                  ... f3ed48b92fc3a93e1cb452da2454bf9d8e117b35
                                  ... ")

cc.write(ds, 'flight_initial', overwrite='True')
cc.write(df, 'flight_actual', overwrite='True')

dl = pd.read_csv('../SRC/labels_Initial.csv')
dt = pd.read_csv('../SRC/labels_Actual.csv')
cc.write(dl, 'charges_initial', overwrite='True')
cc.write(dt, 'charges_actual', overwrite='True')

#Automatic opening of the Carto DB Europe Map
webbrowser.open('https://annaaru.carto.com/builder/ab6706fd-72db
    ... -4631-a134-feed1b9de7ca/embed')


#Creation of a Summary_EN-ROUTE-CHARGES. There is specified
    ...  the cost per country, per initial or actual flight
    ... and totals
myFile = open('../En-Route-Charges/{0}/{1}/Summary_Charges_{2}.csv'.
    ... format(model, airline, flight), 'w')
with myFile:
    writer = csv.writer(myFile)
    writer.writerows(CsvData)
```

Scenario.py

```
#Scenario function contains the two principals Classes of
    ... this script; the objetive is to compute the
    ... environment working, airspace scenario.
import pickle
import pandas
import csv
from shapely.geometry import Point
from shapely.geometry import LineString
from shapely.geometry import Polygon

#It is defined the Class Airspace to read the .pkl airspace
    ...  files and then, convert in Polygons.
class Airspace:
    def __init__(self):
```

```
self.polys = []

with open('SpainPoints.pkl', 'rb') as input:
    self.SectorSpain = pickle.load(input)
self.polySpain = Polygon(self.SectorSpain)

with open('PortugalPoints.pkl', 'rb') as input:
    self.SectorPortugal = pickle.load(input)
self.polyPortugal = Polygon(self.SectorPortugal)

with open('CanariasPoints.pkl', 'rb') as input:
    self.SectorCanarias = pickle.load(input)
self.polyCanarias = Polygon(self.SectorCanarias)

with open('SantaMariaPoints.pkl', 'rb') as input:
    self.SectorSantaMaria = pickle.load(input)
self.polySantaMaria = Polygon(self.SectorSantaMaria)

with open('FrancePoints.pkl', 'rb') as input:
    self.SectorFrance = pickle.load(input)
self.polyFrance = Polygon(self.SectorFrance)

with open('ItalyPoints.pkl', 'rb') as input:
    self.SectorItaly = pickle.load(input)
self.polyItaly = Polygon(self.SectorItaly)

with open('MaltaPoints.pkl', 'rb') as input:
    self.SectorMalta = pickle.load(input)
self.polyMalta = Polygon(self.SectorMalta)

with open('UKPoints.pkl', 'rb') as input:
    self.SectorUK = pickle.load(input)
self.polyUK = Polygon(self.SectorUK)

with open('IrlandPoints.pkl', 'rb') as input:
    self.SectorIrland = pickle.load(input)
self.polyIrland = Polygon(self.SectorIrland)

with open('BelgiumPoints.pkl', 'rb') as input:
    self.SectorBelgium = pickle.load(input)
self.polyBelgium = Polygon(self.SectorBelgium)

with open('GermanyPoints.pkl', 'rb') as input:
    self.SectorGermany = pickle.load(input)
self.polyGermany = Polygon(self.SectorGermany)

with open('NetherlandsPoints.pkl', 'rb') as input:
    self.SectorNetherlands = pickle.load(input)
```

```
        self.polyNetherlands = Polygon(self.SectorNetherlands)

        with open('AustriaPoints.pkl', 'rb') as input:
            self.SectorAustria = pickle.load(input)
        self.polyAustria = Polygon(self.SectorAustria)

        with open('KovenhavenPoints.pkl', 'rb') as input:
            self.SectorKovenhaven = pickle.load(input)
        self.polyKovenhaven = Polygon(self.SectorKovenhaven)

        with open('NorwayPoints.pkl', 'rb') as input:
            self.SectorNorway = pickle.load(input)
        self.polyNorway = Polygon(self.SectorNorway)

        with open('SwedenPoints.pkl', 'rb') as input:
            self.SectorSweden = pickle.load(input)
        self.polySweden = Polygon(self.SectorSweden)

        with open('SwitzerlandPoints.pkl', 'rb') as input:
            self.SectorSwitzerland = pickle.load(input)
        self.polySwitzerland = Polygon(self.SectorSwitzerland)

    #Using get_countries() function and taking a flight as
        ... the input, the output is the different airspaces
        ... that this flight crosses (route)
    def get_countries(self, flight):
        Route = []
        for point in flight:
            if self.polySpain.contains(point):
                Route.append('LE')
            if self.polyPortugal.contains(point):
                Route.append('LP')
            if self.polyCanarias.contains(point):
                Route.append('GC')
            if self.polySantaMaria.contains(point):
                Route.append('AZ')
            if self.polyFrance.contains(point):
                Route.append('LF')
            if self.polyItaly.contains(point):
                Route.append('LI')
            if self.polyMalta.contains(point):
                Route.append('LM')
            if self.polyUK.contains(point):
                Route.append('EG')
            if self.polyIrland.contains(point):
                Route.append('EI')
            if self.polyBelgium.contains(point):
                Route.append('EB')
```

```
        if self.polyGermany.contains(point):
            Route.append('ED')
        if self.polyNetherlands.contains(point):
            Route.append('EH')
        if self.polyKovenhaven.contains(point):
            Route.append('EK')
        if self.polyNorway.contains(point):
            Route.append('EN')
        if self.polySweden.contains(point):
            Route.append('ES')
        if self.polyAustria.contains(point):
            Route.append('LO')
        if self.polySwitzerland.contains(point):
            Route.append('LS')

    i = 0
    while i <= (len(Route)-1):
        j = i + 1
        while j <= (len(Route)-1):
            if Route[i] == Route[j]:
                del(Route[j])
                j = j - 1
            j = j + 1
        i = i + 1
    return Route

# Using get_polys() function and taking the route
    ... previously computed as a input, the output is a
    ... vector that defines the airspaces acronyms.
def get_polys(self, route):
    self.polys = []
    for string in route:
        if string == 'LE':
            self.polys.append(self.SectorSpain)
        if string == 'GC':
            self.polys.append(self.SectorCanarias)
        if string == 'LP':
            self.polys.append(self.SectorPortugal)
        if string == 'AZ':
            self.polys.append(self.SectorSantaMaria)
        if string == 'LF':
            self.polys.append(self.SectorFrance)
        if string == 'LI':
            self.polys.append(self.SectorItaly)
        if string == 'LM':
            self.polys.append(self.SectorMalta)
        if string == 'EG':
            self.polys.append(self.SectorUK)
```

```
        if string == 'EI':
            self.polys.append(self.SectorIrland)
        if string == 'EB':
            self.polys.append(self.SectorBelgium)
        if string == 'ED':
            self.polys.append(self.SectorGermany)
        if string == 'EH':
            self.polys.append(self.SectorNetherlands)
        if string == 'EK':
            self.polys.append(self.SectorKovenhaven)
        if string == 'EN':
            self.polys.append(self.SectorNorway)
        if string == 'ES':
            self.polys.append(self.SectorSweden)
        if string == 'LO':
            self.polys.append(self.SectorAustria)
        if string == 'LS':
            self.polys.append(self.SectorSwitzerland)


#This class take the flight Excel file to compute the
    ... latitude and longitude of each flight point. The
    ... output is a Linestring
# to use this vector in future calculations efficiently.
class Flight:
    def __init__(self):
        self.listFlight = []
        self.FlightCoord = []
        self.Route = []

    def get_flight_coordinates(self, n, sheet):
        data = pandas.read_excel(n, sheetname=sheet)
        lat = (data['Latitude']).tolist()
        lon = (data['Longitude']).tolist()
        Data = [['lat', 'lon']]
        grad = []
        minut = []
        secon = []
        grad2 = []
        minut2 = []
        secon2 = []
        latitude = []
        longitude = []
        for i in range(len(lat)):
            grad.append(lat[i]//10000)
            minut.append((lat[i] % 10000)//100)
            secon.append(lat[i] % 100)
            latitude.append(grad[i] + minut[i]/60 + secon[i]/3600)
```

```
        if lon[i]<0:
            grad2.append(abs(lon[i]) // 10000)
            minut2.append((abs(lon[i]) % 10000) // 100)
            secon2.append(abs(lon[i]) % 100)
            longitude.append(-grad2[i] - minut2[i]/60 - secon2[i
                ... ]/3600)
        else:
            grad2.append(lon[i] // 10000)
            minut2.append((lon[i] % 10000) // 100)
            secon2.append(lon[i] % 100)
            longitude.append(grad2[i] + minut2[i] / 60 + secon2[i]
                ... / 3600)
        Data.append([latitude[i], longitude[i]])
        p1 = Point([latitude[i], longitude[i]])
        self.listFlight.append(p1)
    self.FlightCoord = LineString(self.listFlight)

    #The flight coordinates are exported as csv to send
        ... to CartoDB Platform.
    myFile = open('{0}.csv'.format(sheet), 'w')
    with myFile:
        writer = csv.writer(myFile)
        writer.writerows(Data)
```

Intersection

```
#The principal objetive of this class is to achieve the
    ... entry and exit crossing points for each
#flown airspace.
import shapely
from shapely.geometry import MultiPoint


class Intersection:

    def __init__(self):
        self.intersectionPoints = []
        self.enRoutePoints = []

    #First, this function create a vector (
        ... intersectionPoints) using function intersection
        ... from shapely library. This vector contains the
        ... intersection
    #points of the flight.
    def get_intersection(self, flight, polys, route, fcoords):
        i = 0
        self.intersectionPoints = []
        self.enRoutePoints = []
        while i <= (len(polys) - 1):
```

```
        self.intersectionPoints.append(route[i])
        point = polys[i].intersection(flight)
        if type(point) == shapely.geometry.multipoint.MultiPoint:
            if fcoords[0].x > fcoords[len(fcoords)-1].x:
                a = point[0]
                b = point[1]
                point = shapely.geometry.multipoint.MultiPoint([b, a
                    ... ])
        if i < 1:
            if type(point) == shapely.geometry.multipoint.
                ... MultiPoint:
                n = len(point) - 1
                self.intersectionPoints.append(point[n])
            else:
                self.intersectionPoints.append(point)
        elif i >= (len(polys)-1):
            if type(point) == shapely.geometry.multipoint.
                ... MultiPoint:
                self.intersectionPoints.append(point[0])
            else:
                self.intersectionPoints.append(point)
        else:
            if type(point) == shapely.geometry.multipoint.
                ... MultiPoint:
                self.intersectionPoints.append(point[0])
                n = len(point) - 1
                self.intersectionPoints.append(point[n])
            else:
                self.intersectionPoints.append(point)
        i = i+1

#Finally, this function adds to the vector
    ... intersectionPoints the departure point and
    ... arrival point that they are not intersected with
    ... any airspace.
def get_en_route_points(self, flight):
    self.enRoutePoints.append(self.intersectionPoints[0])
    self.enRoutePoints.append(flight[0])
    i = 1
    while i <= (len(self.intersectionPoints) - 1):
        self.enRoutePoints.append(self.intersectionPoints[i])
        i = i+1
    i = (len(flight))
    self.enRoutePoints.append(flight[i-1])
```

Ddbb.py

```
#The objetive of Postgresql class is to import different
    ... important parameters from Airplane Solutions
```

```
    ... database.
import psycopg2


class Postgresql(object):
    def __init__(self):
        self.db_param = {'host':'13.95.30.195',
                         'database':'SHOGANAI',
                         'user':'postgres',
                         'password':'S1lv3str301'}

    #get_unit_rates() function call the database to get the
        ... monthly unit rates from Eurocontrol. The unit
        ... rates are determined
    # according to the airspaces defined per a flight (route
        ... )
    def get_unit_rates(self, route):

        listunitrates= []

        try:
            connection = psycopg2.connect(**self.db_param)
            cursor = connection.cursor()

            select_query = """SELECT state_member, unit_rate
                ... FROM aps_unit_rates WHERE state_member IN (
                ... """
            routestring = "'"+route[0]+"'"
            for country in route[1:]:
                routestring = routestring + ",␣'" + country + "'"

            select_query = select_query + routestring + ')'
            cursor.execute(select_query)
            listunitrates = cursor.fetchall()
            connection.commit()

        except (Exception, psycopg2.Error) as error:
            print("Error␣while␣connecting␣to␣PostgreSQL", error)

        finally:
            # closing database connection.
            if connection:
                cursor.close()
                connection.close()
            return listunitrates

    #get_mtom() function call the database to get the mtom
        ... according to the airplane tail number which
```

```
    ... perform a flight.
    def get_mtom(self):

        flight_mtom = 0

        try:
            connection = psycopg2.connect(**self.db_param)
            cursor = connection.cursor()

            select_query = "SELECT␣mtom_tm␣FROM␣aps_aircraft␣WHERE␣
                ... tail_number␣=␣'10222IT'"
            cursor.execute(select_query)
            flight_mtom = cursor.fetchall()
            connection.commit()

        except (Exception, psycopg2.Error) as error:
            print("Error␣while␣connecting␣to␣PostgreSQL", error)

        finally:
            # closing database connection.
            if connection:
                cursor.close()
                connection.close()
            return flight_mtom
```

EnRouteCharge.py

```
#The most important Class is EnRoute. Here, it is defined
    ... the different parameters and functions to calculate
    ... the final en-route charge
# per country and per flight.
import math


class EnRoute:
    def __init__(self):
        self.en_route_charges = []
        self.GreatDist = []
        self.EnRouteFactors = []
        self.Country = []
        self.R = 6376
        self.WeightFactor = []
        self.initiallatitude = []
        self.initiallongitude = []
        self.finallatitude = []
        self.finallongitude = []
        self.TotalDistance = []

    #get_unit() function creates a vector (list_rates) that
```

```
    ... assign each unit rate to an airspace acronym,
    ... with the objetive to organize the unit rates.
def get_unit(self, Unit_rates, Route):
    list_rates = []
    i = 0
    j = 0
    while i <= len(Unit_rates):
        while j <= (len(Route) - 1):
            if Route[j] == Unit_rates[i][0]:
                list_rates.append(Unit_rates[i][1])
                i = 0
                j = j + 1
            else:
                i = i + 1
        if j >= len(Route):
            i = (len(Unit_rates) + 1)
    return list_rates

#get_distance calculate the orthodromic distance between
    ...  the two intersection points
#for each flown airspace.
def get_distance(self, enRoutePoints, list_rates):
    i = 0
    d = []
    self.Country = []
    self.initiallatitude = []
    self.initiallongitude = []
    self.finallatitude = []
    self.finallongitude = []
    self.GreatDist = []
    self.EnRouteFactors = []
    while i <= (len(enRoutePoints)-1):
        self.Country.append(enRoutePoints[i])
        self.initiallatitude.append(enRoutePoints[i + 1].x)
        self.initiallongitude.append(enRoutePoints[i + 1].y)
        self.finallatitude.append(enRoutePoints[i + 2].x)
        self.finallongitude.append(enRoutePoints[i + 2].y)
        i = i + 3

    i = 0
    while i < len(self.Country):
        self.GreatDist.append(self.Country[i])
        self.EnRouteFactors.append(self.Country[i])
        self.EnRouteFactors.append(list_rates[i])

        #Conversion to radians
        ilat = math.radians(self.initiallatitude[i])
        ilon = math.radians(self.initiallongitude[i])
```

```
            flat = math.radians(self.finallatitude[i])
            flon = math.radians(self.finallongitude[i])

            #Latitude and longitude differences
            dlat = flat - ilat
            dlon = flon - ilon

            #Great Circle Distance Calculation
            a = (math.sin(dlat/2))**2 + math.cos(ilat)*math.cos(flat)*(
                ... math.sin(dlon/2))**2
            c = 2*math.asin(min(1, math.sqrt(a)))
            d.append(self.R*c)
            self.EnRouteFactors.append(d[i]/100)
            i = i + 1
        return d

    #Using the unit rates defined previously, mtom and Great
        ...  Circle Distance, the formula to compute the en-
        ... route charge is applied.
    #It return a vector Enroute_charge with the results.
    def get_enroute_charge(self, factors, Mtom):
        i = 0
        j = 0
        Enroute_charge = []
        while i < len(factors):
            Enroute_charge.append(factors[i])
            Unit_rate = float(factors[i+1]/100)
            DistanceFactor = factors[i+2]
            Charge = Unit_rate*DistanceFactor*math.sqrt(Mtom/50)
            Enroute_charge.append(round(Charge, 2))
            i = i + 3
        return Enroute_charge
```

SetLabels.py

```
#The objetive of Labels class is to process the results to
    ... assign them to airspace and flight. Then, the
    ... results will be plotted and exported.
import csv


class Labels:

    def __init__(self):
        self.country = []
        self.points = []
        self.charge = []
        self.distance = []
        self.aplication = []
```

```
    self.myCost = [['Flight', 'Category', 'Country', 'Cost']]

#set_labels() function defines different parameters in
    ... order to classify them orderly, according to
    ... country, coordinates and charge of an specific
#airspace intersection.
def set_labels(self, enroutepoints, charges, d):
    self.country = []
    self.points = []
    self.charge = []
    self.distance = []
    self.country.append(enroutepoints[0])
    self.points.append(enroutepoints[1])
    self.charge.append('')
    i = 0
    m = 1
    while i < len(enroutepoints):
        if i <= (len(enroutepoints)-4):
            self.country.append(enroutepoints[i])
            self.points.append(enroutepoints[i+2])
            self.charge.append(charges[m])
        else:
            self.country.append(enroutepoints[i])
            self.points.append(enroutepoints[i+2])
            self.charge.append(charges[m])
        i = i + 3
        m = m + 2

    distance2 = []
    j = 0
    sum = 0
    while j < len(d):
        sum = d[j] + sum
        j = j + 1

    n = 1
    distance2.append(d[0])
    self.distance.append(round(((distance2[0]/sum)*100), 2))
    while n < len(d):
        distance2.append(distance2[n-1] + d[n])
        self.distance.append(round((distance2[n]/sum)*100, 2))
        n = n + 1

#To export the data to CartoDB platform, it is necessary
    ...  to create a csv file with the previous processed
    ...  data.
def create_csv(self, enroutepoints, sheet, flight):
    i = 0
```

```
myData = [['country', 'lat', 'lon', 'wiglet']]
while i < len(self.charge):
    if self.country[i] == 'LE':
        self.country[i] = 'Spain␣Continental'
    if self.country[i] == 'GC':
        self.country[i] = 'Spain␣Canarias'
    if self.country[i] == 'LP':
        self.country[i] = 'Portugal␣Lisboa'
    if self.country[i] == 'AZ':
        self.country[i] = 'Portugal␣S␣M'
    if self.country[i] == 'LF':
        self.country[i] = 'France'
    if self.country[i] == 'LI':
        self.country[i] = 'Italy'
    if self.country[i] == 'LM':
        self.country[i] = 'Malta'
    if self.country[i] == 'EG':
        self.country[i] = 'GBP␣United␣Kingdom'
    if self.country[i] == 'EI':
        self.country[i] = 'Ireland'
    if self.country[i] == 'EB':
        self.country[i] = 'Belg.-Luxembourg'
    if self.country[i] == 'ED':
        self.country[i] = 'Germany'
    if self.country[i] == 'EH':
        self.country[i] = 'Netherlands'
    if self.country[i] == 'EK':
        self.country[i] = 'Denmark'
    if self.country[i] == 'EN':
        self.country[i] = 'Norway'
    if self.country[i] == 'ES':
        self.country[i] = 'Sweden'
    if self.country[i] == 'LO':
        self.country[i] = 'Austria'
    if self.country[i] == 'LS':
        self.country[i] = 'Switzerland'
    i = i + 1
string1 = '{0}␣-␣{1}{2}'.format(self.country[0], 'DEPARTURE␣
    ... APT', self.charge[0])
myData.append([string1, self.points[0].x, self.points[0].y])
sum = 0
j = 1
n = 0

# Also, the results are defined in order to plot in a
    ...  Pop-Up Legend in CartoDB Platform for each
    ... intersection point, that display the cost per
    ... country,
```

```
# the acumulative cost and the % of flight performed
    ... for the moment.
while j < (len(self.country)):
    sum = sum + int(self.charge[j])
    if j >= (len(self.country) - 1):
        string = '{0}␣-␣{1}:␣{2}\u20ac␣\n␣\␣Cost␣since␣
            ... departure:␣{3}\u20ac␣\n␣\␣\u0025Flight␣progress
            ... :␣{4}\u0025'.format(self.country[j], 'ARRIVAL␣
            ... APT', self.charge[j], str('{:,}'.format(sum)),
            ...  self.distance[n])
        string2 = '{0}:␣{1}\u20ac'.format(self.country[j],
            ... self.charge[j])
        myData.append([string, self.points[j].x, self.points[j
            ... ].y, string2])
        self.myCost.append([flight, sheet, self.country[j],
            ... self.charge[j]])
        n = n + 1
    else:
        string = '{0}:␣{1}\u20ac␣\n␣\␣Cost␣since␣departure:␣
            ... {2}\u20ac␣\n␣\␣\u0025Flight␣progress:␣{3}\u0025
            ... '.format(self.country[j], '{:,}'.format(self.
            ... charge[j]), str('{:,}'.format(sum)), self.
            ... distance[n])
        string2 = '{0}:␣{1}\u20ac'.format(self.country[j],
            ... self.charge[j])
        myData.append([string, self.points[j].x, self.points[j
            ... ].y, string2])
        self.myCost.append([flight, sheet, self.country[j],
            ... self.charge[j]])
        n = n + 1
    j = j + 1
self.myCost.append([flight, sheet, 'Total', sum])

#Finally, the final csv file is created to exported
    ... to Carto.
myFile = open('labels_{0}.csv'.format(sheet), 'w')
with myFile:
    writer = csv.writer(myFile)
    writer.writerows(myData)

return self.myCost
```